DOCUMENT RESUME

ED 291 346 IR 013 134

AUTHOR	Ryan, William C., Ed.
TITLE	National Educational Computing Conference Proceedings (8th, Philadelphia, Pennsylvania, June 24-26, 1987).
REPORT NO	ISBN-0-924667-40-0
PUB DATE	Jun 87
NOTE	419p.; For the 1986 conference proceedings, see ED 274 338.
AVAILABLE FROM	International Council on Computers for Education, University of Oregon, 1787 Agate Street, Eugene, OR 97403-1923 ($15.00 plus $2.50 shipping and handling prepaid).
PUB TYPE	Collected Works - Conference Proceedings (021) -- Viewpoints (120) -- Reports - Descriptive (141)
EDRS PRICE	MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS	*Computer Assisted Instruction; *Computer Literacy; Computer Science Education; *Computer Software; Elementary Secondary Education; Higher Education; *Intellectual Disciplines; *Microcomputers; Programing; Special Education; Teacher Education; Videodisks

ABSTRACT

The almost 200 papers and panel, project, and special session reports represented in this collection focus on innovations, trends, evaluations, and research in the use of computers in a variety of educational settings. Of these, the full text is provided for 52 presentations and abstracts for 138. The topics discussed include: use of computers at various levels from K-12 through the university, including inservice teacher training; use of computers in specific subject areas such as business education, science, mathematics, health education, literature, foreign languages, social studies, and English; computers and special education; questions of ethics, equity, and special needs in computer access and use; the teaching of programming; software development and selection; computer literacy; computer support facilities; computer-related activities of the Pennsylvania Department of Education; the computer program at Drexel University; industry/school district cooperation in providing computer education; computer science education; evaluation of the effectiveness of computer-assisted instruction; cognitive skills and processes; and computer simulation. Introductory materials include brief descriptions of several of the cooperating societies that sponsor the National Educational Computing Conference (NECC) as well as lists of the members of the NECC Steering Committee and the NECC '87 Conference Committee. An author index is provided. (EW)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED291346

PROCEEDINGS

National Educational Computing Conference
June 24-26, 1987

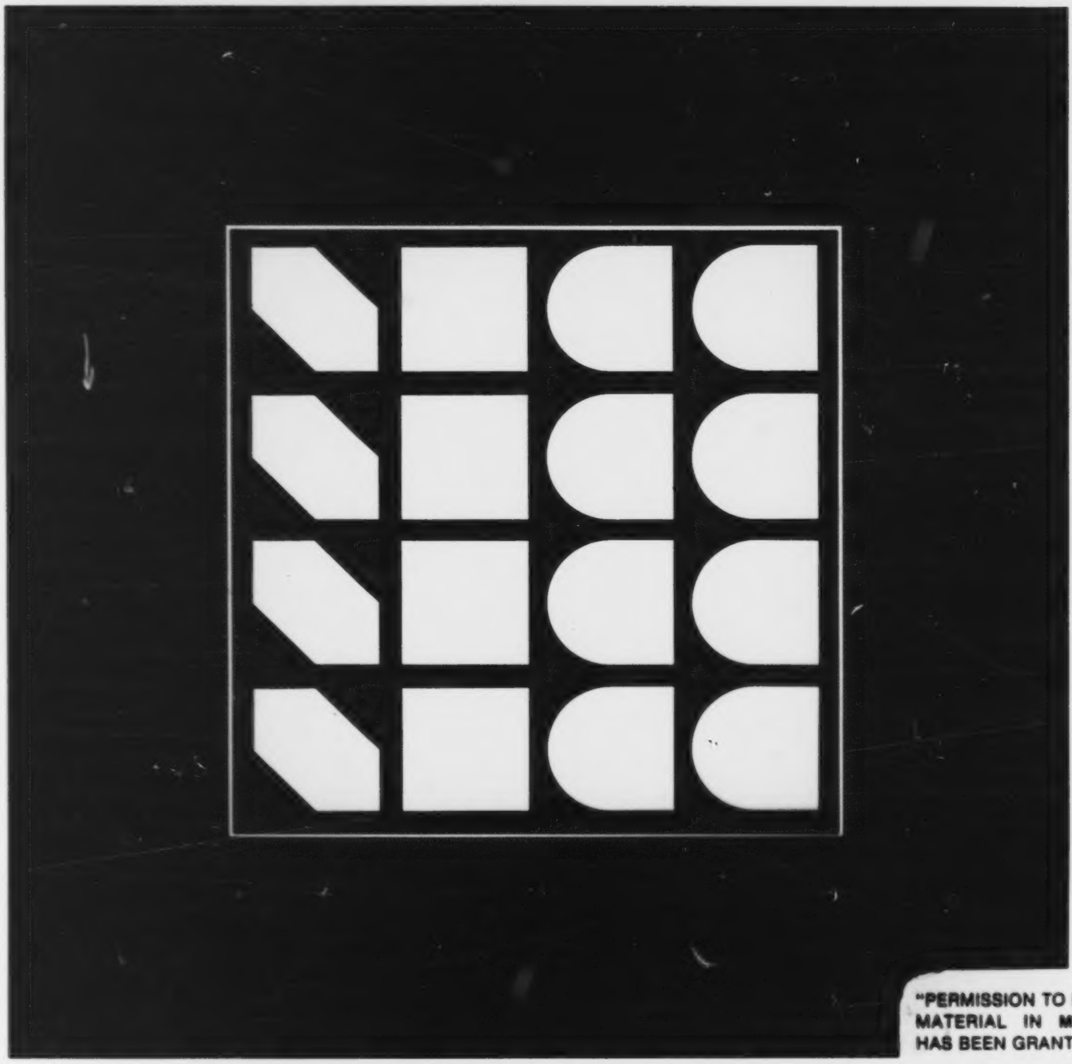
U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

☒ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality

- Points of view or opinions stated in this document do not necessarily represent official OERI position or policy

Philadelphia, Pennsylvania
Host: Temple University



IR013134

"PERMISSION TO REPRODUCE THIS MATERIAL IN MICROFICHE ONLY HAS BEEN GRANTED BY

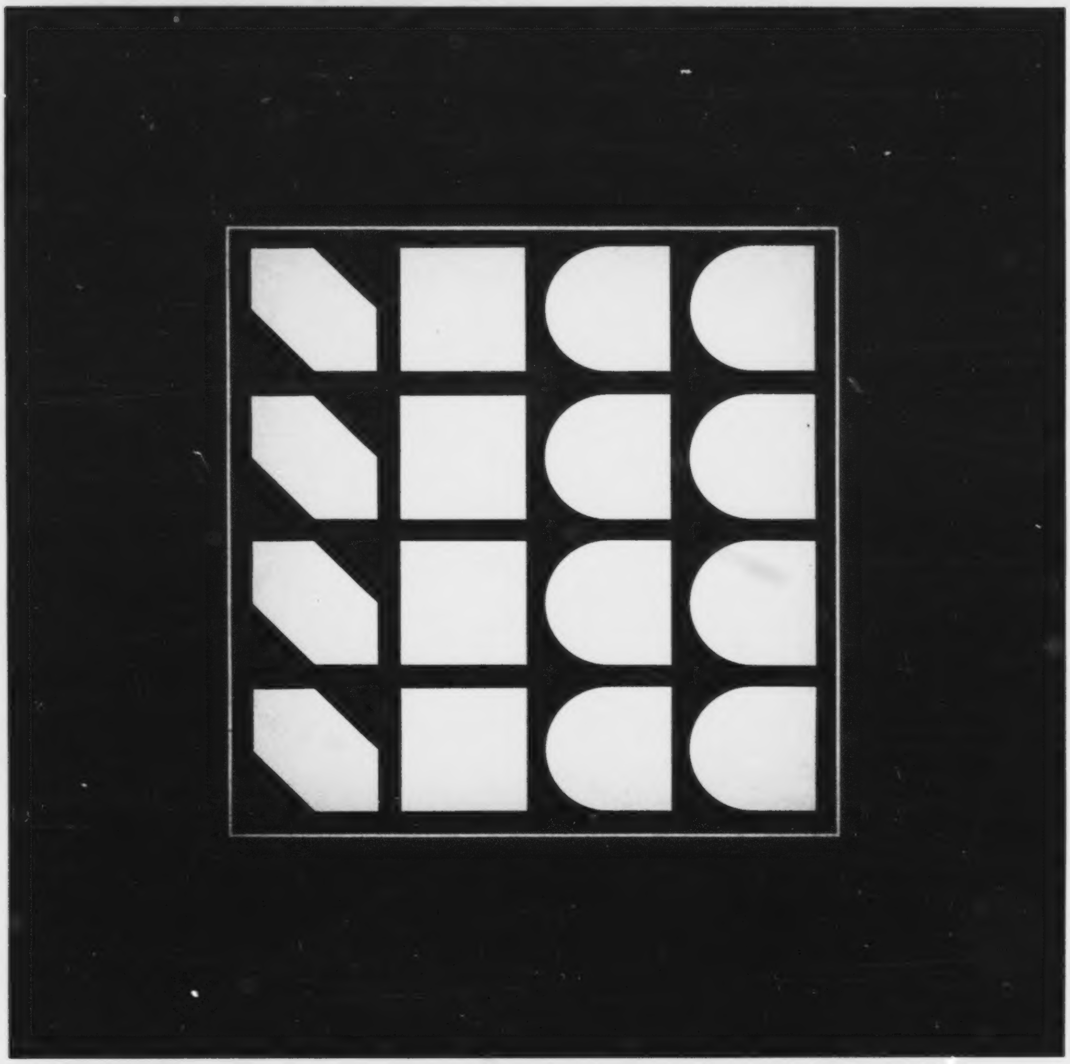
NECC

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

PROCEEDINGS

National Educational Computing Conference
June 24-26, 1987

Philadelphia, Pennsylvania
Host: Temple University



EDITED BY

William C. Ryan
Swarthmore College

PROCEEDINGS OF NECC '87

International Standard Book Number: 0-924667-40-0

Copyright © 1987: NECC

Published by:
International Council on Computers for Education
University of Oregon
1787 Agate Street
Eugene, OR 97403-1923
for
National Educational Computing Conference 1987
June 1987

Cover Design by Barbara Hoffert

NATIONAL EDUCATIONAL COMPUTING CONFERENCE

COOPERATING SOCIETIES

- American Educational Research Association (AERA)
- American Society for Engineering Education/Computers in Education Division (ASEE/CoED)
- Association for Computers and the Humanities (ACH)
- Association for Computing Machinery/Special Interest Groups on:
  - Computer Science Education (SIGCSE)
  - Computer Uses in Education (SIGCUE)
  - University and College Computing Services (SIGUCCS)
- Association of Small Computer Users in Education (ASCUE)
- Educational Computing at Minority Institutions (ECMI)
- EDUCOM/EDUNET
- International Association for Computing in Education (IACE)
- International Council for Computers in Education (ICCE)
- IEEE Computer Society
- Society for Computer Simulation (SCS)

NECC STEERING COMMITTEE

Doris K. Lidtke
Chair
Towson State University

William C. Ryan
Vice-Chair
Swarthmore College

Ted Sjoerdsma
Treasurer
Washington and Lee University

Diana Harris
Secretary

Ronald E. Anderson
ACM/SIGCUE
University of Minnesota

Anita Best
ICCE
University of Oregon

Gerald L. Engel
Past Chair, Steering Committee
University of Connecticut at Stamford

Priscilla Jane Huston
ACM/SIGUCCS
Rice University

Nancy M. Ide
ACH
Vassar College

Jesse C. Lewis
ECMI
Norfolk State University

James Lubkin
ASEE/CoED
Michigan State University

Michael Mulder
IEEE Computer Society
University of Portland

Dan Oberst
EDUCOM/EDUNET

Jim Poirot
Chair, NECC '88
North Texas State University

Nancy Roberts
SCS
Lesley College

W. Waldo Roth
ASCUE
Taylor University

Thomas Saterfiel
AERA
Mississippi State Department of Education

Dennis Spuck
IACE
University of Houston

A. Joe Turner
ACM/SIGCSE
Clemson University

Susan M. Zgliczynski
Chair, NECC '86
University of San Diego

NECC '87 CONFERENCE COMMITTEE

Conference Chair
Frank Friedman
Temple University

Associate Chairs and Conference Registration
Irene Tasca
Temple University
Rev. Richard Antonucci
Archdiocese of Philadelphia

Conference Treasurer
Susan Foster
Temple University

Assistant Treasurer and Registration Records
Helen Holzbaur
Temple University

Registration Records
Sister Joseph Kieran McAdams

Special Events
Myrna Josephs
AT&T

Student Assistants Coordinator
Catherine Owens
Temple University

Local Arrangements
Patricia Carroll
CertainTeed Corp.

Local Arrangements Committee
Frank Betts
Joseph Ducette
Jessica Kahn
Arlene Kramer
Calvin Nodine
Laurie Shtier

Publicity Chair
Gayle J. Yaverbaum
Temple University

Publicity Associate Chair
Ruth Schwartz
Temple University

Program Chair and Project Presentations
Jean B. Rogers
Stanford University

Special Sessions and Workshops
Anita Best
ICCE

Papers
Linda D. Rathje
ICCE

Society Sessions
A. Joe Turner
Clemson University

Spotlight on the Delaware Valley
Holly Jobe
Montgomery County Pa. Intermediate Unit

Industrial Coordinator
Sandra Morris
RCA

Exhibits
Paul Katz
Nelda K. Owsiak
University of Oregon Continuation Center

ASSOCIATION FOR COMPUTERS AND THE HUMANITIES

The Association for Computers and the Humanities (ACH) is an international organization devoted to encouraging the development and use of computing techniques in humanities research and education. Traditionally, ACH has fostered computer-aided research in literature and language, history, philosophy, anthropology, and related social sciences as well as computer use in the creation and study of art, music, and dance. As computing applications in the humanities have developed and broadened in the 1980s, the Association has expanded its scope to include areas from word processing to computer-assisted instruction in composition, language, history, philosophy, and anthropology, as well as computational linguistics and cognitive science, which overlap increasingly with work in the area of humanities computing.

Founded in 1977, ACH is the primary professional society for humanists who are involved or interested in any aspect of humanities computing. The Association provides a forum for continuing communication about humanities computing and strives to meet the needs of those who want to gain familiarity with both existing and potential applications of computers in humanities disciplines.

The heart of ACH is its quarterly newsletter, which covers the activities of the Association and its members and includes articles on various areas within humanities computing, news of projects and conferences of interest to ACH members, and reports on activities of governmental agencies and other organizations that affect computer-aided humanities research.

Computers and the Humanities, published by Paradigm Press, is a quarterly journal devoted to scholarship in the field of humanities computing. Subscription to *CHum* is included in the ACH membership fee.

ACH sponsors the bi-annual International Conference on Computers and the Humanities (ICCH), held in odd-numbered years, which brings together scholars from around the world to report on research activities and software and hardware developments in the field. Recently, ACH began to sponsor conferences and workshop on specialized topics in humanities computing, held in even-numbered years.

ACM SPECIAL INTEREST GROUP ON COMPUTER SCIENCE EDUCATION

SIGCSE became a special interest group of ACM in 1970. It currently consists of over 2000 members from the educational, industrial, and governmental communities interested in various aspects of computer science education. SIGCSE has goals of encouraging and assisting in the development of effective academic programs and courses in computer science and promoting research in computer science education.

The following are objectives of SIGCSE:

1. To provide a continuing forum for the discussion of common problems among educators and other computer scientists through organized meetings and symposia.
2. To publish a bulletin at least quarterly containing information aimed specifically at those interested in computer science education.
3. To work closely with the Education Board of ACM to insure implementation of effective education programs by the Association.

ACM SPECIAL INTEREST GROUP ON COMPUTER USES IN EDUCATION

The Special Interest Group on Computer Uses in Education (SIGCUE) of the Association for Computing Machinery (ACM) provides a forum for the discussion of ideas, methods, and policies related to all aspects of computers in the educational process. Established in 1969 its membership (over 1,500 persons) comes from many countries and numerous, diverse institutions.

SIGCUE publishes a newsletter, formerly called *SIGCUE Bulletin*, that last year was named *SIGCUE Outlook*. SIGCUE also sponsors and organizes technical sessions at ACM annual meetings, the National Educational Computing Conferences, and other national meetings of interest to its members. A recent SIGCUE project consisted of a study group on national educational computer policy issues. This culminated in the publication last year by ACM of an edited volume entitled *National Educational Computer Policy Alternatives*.

SIGCUE Chairman, Robert P. Taylor of Columbia Teachers College, identified SIGCUE's major goals as (1) publishing a current, substantive bulletin, (2) cooperating with other special interest groups or educational societies to promote attention to educational computing issues, and (3) helping to bring the technical expertise within ACM to bear upon educational computing generally.

ACM SPECIAL INTEREST GROUP ON COLLEGE & UNIVERSITY COMPUTING SERVICES

SIGUCCS provides a forum for those involved in providing computing services on a college or university campus. The topics addressed by SIGUCCS include managing campus computing, computing as it relates to the overall goals of the institution, and the state-of-the-art in various types of college and university computing services. Membership in SIGUCCS will help an individual to keep current with the world of collegiate computing services, and provides opportunities to discuss and share ideas and experiences with others.

Two annual conferences are regular activities of SIGUCCS. The Computer Center Management Symposium addresses the many aspects of managing computing on campus. This includes hardware, software, planning, finances, and personnel, to name a few. The User Services Conference deals more directly with the delivery of particular services to the higher education community.

In other projects, SIGUCCS offers a Peer Review of the university computing function. Upon request of the computer center director, members of SIGUCCS will formally analyze and comment on different areas of your campus computing function. SIGUCCS also publishes a quarterly Newsletter. We consider the Newsletter our most important form of communication as it reaches all members and is subscribed to by numerous university computing centers. Conference proceedings are published either as separate documents or as part of the Newsletter itself.

THE ASSOCIATION OF SMALL COMPUTER USERS IN EDUCATION

The Association of Small Computer Users in Education (ASCUE) is a group of small colleges who work toward continued quality computer education and computer assistance in both administrative and academic areas. The purposes of the organization are: 1) to encourage appropriate uses of computing equipment and techniques for the improvement of its member institutions; 2) to supply its members with information on the most current data processing methods; 3) to assist them in solving individual problems; 4) to cooperate with them in the utilization of the various small computers; 5) to cooperate with manufacturers, distributors, and suppliers in establishing and maintaining proper technical standards and in meeting new needs for special devices and systems.

EDUCATIONAL COMPUTING IN MINORITY INSTITUTIONS

ECMI is an organization run by a steering committee representing institutions whose student body reflects a large identifiable minority population. The objectives of ECMI are:

1. Computer literacy: To create among the faculty and administrators of the minority institutions an awareness and understanding of the strengths and weaknesses, uses and misuses, advantages and disadvantages, feasibility, practicability and limitations of computer applications in all aspects of society, including education.

2. Educational Computing: To narrow the gap which exists between the faculties in minority and non-minority institutions with respect to educational computing know-how and access.

3. Research Computing: To improve the computing facilities available to faculty of minority institutions for research purposes, particularly in those institutions offering graduate programs.

4. Technical assistance—consultants: To provide expert and impartial technical assistance to academic administrators of minority institutions on all phases of academic computing (instruction and research).

5. Education programs in the computer sciences: To improve the offerings of courses and degree programs in the computer sciences at minority institutions at all levels (e.g., introductory courses, minors, 2-year degree programs, 4-year degree programs, continuing education, graduate programs).

6. Computing facilities: To improve both quality and quantity of computing facilities available at minority institutions, because experience in the non-minority institutions has shown that an adequately staffed and equipped computer center for academic computing is essential to the success of previously stated objectives.

7. Direct student assistance: To increase the availability of minority staff for the computer centers and computer science education programs of minority institutions.

8. The need for a comprehensive program: To facilitate co-ordination and equitable distribution of funded activities to qualified institutions, associations, etc.

EDUCOM

EDUCOM is a nonprofit consortium of colleges, universities, and other nonprofit institutions founded in 1964 to facilitate the introduction, use, and management of information technology. Through direct services and cooperative efforts, EDUCOM assists its members and provides leadership to the higher education community.

Over 500 institutions participate in EDUCOM's activities: General Membership (conferences, publications, regional seminars, and discount program), Consulting Group, EDUNET (directories, documentation, account initiation, and billing for remote users of 19 major university electronic mail systems), Planning Model Activities (software - EFPM, consulting, and support; Higher Education Data Sharing Service - HEDS for interinstitutional comparison and analysis), Computer Literacy Project (meeting the needs of non-specialists in the use of computers, identification and transfer of successful initiatives in higher education).

INTERNATIONAL ASSOCIATION FOR COMPUTING IN EDUCATION

The International Association for Computing in Education (IACE), formerly the Association for Educational Data Systems (AEDS), continues to serve the educational computing field in three major areas: administrative, instructional, and research. It is developing international ties with the ministries of education of six nations and also expanding its individual membership internationally. In addition, there is an increased emphasis on the administrative and instructional uses of microcomputers in higher education. Having weathered recent financial difficulties with the help of a grant from Epson America, Inc., the backlog of the research Journal issues are now being produced, and the publication of the IACE Newsletter has resumed. In addition, IACE has entered into a management arrangement with the American Educational Research Association (AERA) to greatly reduce administrative costs. As a result of this arrangement, IACE will have its future annual conferences in conjunction with the AERA annual conferences, and the first IACE/AERA co-conference will be held April 5-9, 1988 in New Orleans. The theme of that conference will be Educational Computing: Practice, Policy, and Research. With the name change, new management arrangement with AERA, new membership structure, and new areas of emphasis, IACE will continue its 26-year tradition of serving the field of educational computing and looks forward to establishing cooperative efforts with other associations in the field.

THE INTERNATIONAL COUNCIL FOR COMPUTERS IN EDUCATION

The International Council for Computers in Education, ICCE, is a non-profit educational organization, with 14,000 Individual Members and 52 Organization and Associate Members.

ICCE publishes *The Computing Teacher*, C.A.L.L. Digest, and many booklets for persons interested in the instructional use of computers at the precollege level. These publications emphasize teaching about computers, teaching using computers, teacher education, and the impact of computers on curriculum.

ICCE has a substantial and growing professional outreach program. Four major components currently include:

1. Organization and Associate Members. Members publish newsletters and/or journals, hold conferences, and directly interact with their own members.

2. Professional Staff. ICCE has a professional staff who write, edit, participate in conferences, process orders, consult by phone or mail, etc. *The Computing Teacher* is put together by a full in-house production staff.

3. Ad Hoc Committees. Such a committee recently created the "ICCE Policy Statement on Software Copyright" and "Code of Ethical Conduct for Computer Using Teachers".

4. Special Interest Groups. ICCE has organized special interest groups for computer coordinators, teachers of educators, computer science educators, and videodisc-microcomputer users.

THE COMPUTER SOCIETY OF THE IEEE

The Computer Society is the world's largest association of computing professionals, with a total membership of approximately 90,000 computer scientists, computer engineers, and allied professionals. Society membership is open to IEEE members, associate members, and student members and to non-IEEE members who qualify for affiliate membership. An affiliate member is a person who has achieved status in his or her chosen field of specialization and whose interests focus in the computing field.

Every Computer Society member receives *COMPUTER*, a monthly magazine of general interest to computing professionals which also covers society news and events. Five specialized magazines and three journals are also available to society members as optional subscriptions and to nonmembers, libraries, and organizations. Magazines published by the Computer Society include *IEEE COMPUTER GRAPHICS & APPLICATIONS, IEEE MICRO, IEEE DESIGN & TEST, IEEE SOFTWARE*, and *IEEE EXPERT*. Research-oriented journals include *IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, and *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*. The Computer Society Press publishes nonperiodical literature, including tutorial texts and conference records. Last year 82 new titles were produced, and the society's catalog now contains almost 700 titles.

The society sponsors or cosponsors more than 100 conferences and meetings ranging from workshops and symposia with a few dozen participants to major conferences with many thousands of attendees. Over 30 technical committees offer the opportunity to interact with peers in technical specialty areas, receive newsletters, and conduct conferences and tutorials.

The Computer Society has over 100 local chapters throughout the world, and an additional 100-plus student chapters which provide the opportunity to interact with local colleagues and hear experts discuss technical issues. In addition, tutorials, educational activities, accreditation of computer science and engineering academic programs, the development of standards, and an international electronic mail network all play prominent roles in the society's activities.

THE SOCIETY FOR COMPUTER SIMULATION

SCS is the only technical society devoted primarily to the advancement of simulation and allied technology. It has a worldwide membership and a network of regional councils that covers the United States, Canada, and the United Kingdom.

The Society's original interest in the use of simulation in aerospace and engineering has expanded into many other industries, disciplines, and professions, including applications dealing with scientific, management, social, biological, and environmental problems. Artificial intelligence, robotics, and CAD/CAM are areas that have been specifically recognized as important to SCS members.

There are two major SCS conferences each year. They are the SCS Multiconference and the Summer Simulation Conference (SCSC).

The National Educational Computing Conference thanks the following people for their contribution of knowledge, effort, and time as referees for the papers submitted for presentation.

Richard Adams
Ronald Anderson
Lynne Anderson-Inman
Carolyn Anderson-Simmons
John Arch
Richard Austing
Bill Baird
Robert Banaugh
Bruce Barnes
Naomi Baron
Nettie Bartel
George Bass
Pat Bauer
Barbara Sue Beall
James Beck
Henry Becker
Anita Best
Julie Bichteler
Rick Billstein
Ruthie Blankenbaker
William Blubaugh
Arlene Borthwick
Joyce Brennan
Gene Brockmeyer
Billie Brown
Kristin Brown
Tom Burnett
Doris Carey
Regan Carey
Cathy Carney
Richard Carter
N. John Castellan, Jr.
Christin Chaille
Sylvia Charp
James Clark
Bob Clausen
Doug Clements
Richard Close
Ronald Cody
Betty Collis
John Colvin
Leslie Conery
Frank Connolly
Sheila Cory
George Culp
Robert Cupper
Nell Dale
John Dalphin
Herbert Dershem
Jean Donham
Karen Duncan
J. Michael Dunlap
B. G. Dunn
Alan Dunn
Gerald Engel
Carolyn Erickson
Linda Ettinger
Selby Evans
Frederick Fink
Geoffrey Fletcher
Dexter Fletcher
David Flowers
Helen Frye
F. Paul Fuhs
David Game
Ricardo Garcia
Judith Gersting
Allen Glenn
Shelly Gordon
Kay Gore
Harold Grossman
Kurt Gustafson
John Hamblen
Michael Haney
Larry Hannah
David Hata
Henry Heikkinen
Shirley Hill
Saundra Hopkins
Nancy Ide
Heidi Imhof
Truman Jackson
Carolyn Jacobi
Larry Jehn
Jerry Johnson
Dale Johnson
Vincent Jones
Beverly Jones
Tony Jongejan
Charles Kelemen
Henry Kepner, Jr.
Phil Kessinger
Joseph Kieran McAdams
Joyce Killian
Carol Klenow
Marvin Koontz
Barbara Kurshan
Glen Langdon
John Lawson
Laura Leventhal
Cliff Lind
Joyce Little
Erica Lodish
William Lyle
Harold Mac Dermot
David Maharry
Gary Marchioni
Stephen Marcus
Kathleen Martin
Judi Mathis
Carla Mathison
Kathy Maury
Jim McCauley
Donald McClain
John McGregor
Diane Means
Sam Miller
Doris Mimnaugh
Keith Mitchell
Andrew Molnar
Margaret Moore-Neiss
Michael Moshell
David Moursund
Barbee Mynatt
Gary Nave
Barbara-Jo Nivitski
David Novick
Larry Oliver
Janet Parker
Nancy Parks Sopp
John Peck
Phil Piele
James Poirot
Linda Polin
Kathy Pon
Anne Porter
Pat Powers
Judy Powers
Gerald Rambally
Arthur Rathje
Doris Ray
Jon Rickman
David Rine
Robert Riser
Nancy Roberts
A. L. Roecks
Jean Rogers
Leroy Roquemore
Donald Ross
Greg Sales

(continued)

NECC Reviewers (continued)

Patrick Salsich
Mary Ellen Sanders
Karon Schaack
Charlotte Scherer
Helen Schwartz
Robert Seidman
Robert Shostak
David Smith
Richard Smith
Philip Spelt
Dennis Spuck
Sally Steadman
Dennis Stevenson
Antonia Stone
Neal Strudler
Gail Sullivan
Elliot Tanis
Marianne Teague
Robert Thomson
Shirley Torgerson
Stan Trollip
Sam Tumolo
Joe Turner
Scott Turner
John VanIwaarden
Henry Walker
Dave Weaver
John Wedman
Fred Weingarten
Keith Wetzel
Brent Wholeben
Frank Wojcik
Billy Wood
Peter Worland
William Wresch
Gayle Yaverbaum
Patti Zembrosky-Barkin
Allen Ziebur

FOREWORD

From both an educational and a technical perspective, we live in very exciting times. As each day passes, both our professional and personal lives become more and more affected by the computer. With increasingly powerful yet less expensive computers and a vast storehouse of easy to use applications software, we are well into an age in which we can extend and transform human capability.

This revolution is as apparent in education as in every other field of endeavor. As educators and computer scientists we find it necessary to keep up with rapid technological development and to assess the impact of this technology on the social and educational environment in which we live. Understanding technology is clearly not enough; we must continually undertake to separate the effective and productive uses of computers from those which are non-productive, wasteful, or even destructive.

Our goal for NECC '87 and this Proceedings is to provide a report on innovations, trends, evaluations, and research in the uses of computers in a variety of disciplines and levels of education. It is hoped that the spirit of NECC '87 will continue to serve you as an inspiration for professional and personal development.

Of course, no conference would be possible without considerable work and cooperation on the part of a large number of people, and NECC '87 is certainly no exception. The high quality of this year's NECC is the result of long-term, sustained efforts by Father Richard Antonucci, Patricia Carroll, Susan Foster, Holly Jobe, Myrna Josephs, Sister Joseph Kieran McAdams, Cal Nodine, Cathy Owens, Jean Rogers, Irene Tassa, and Gayle Yaverbaum.

Other who have made substantial contributions include Donna Baglio, Frank Betts, Helen Holzbaur, Jessica Kahn, Arlene Kramer, Steve Longo, Sandra Morris, Bill Ryan, Ruth Schwartz, Laurie Shtier, Jerry Snyder, and Joe Turner.

I would also like to thank Paul Katz and Nelda Owsiak of the University of Oregon Continuation Center for their work with the exhibits and Linda Rathje of ICCE for her assistance in handling Conference papers.

I am indebted to Bobbie Zucker for the volumes of great advice and guidance that she has provided over the past 3 1/2 years, and to Anita Best of ICCE for helping to coordinate what at times seemed just too many activities. I also want to thank Doris Lidtke, chairman of the NECC Steering Committee, and the rest of her Committee, Diana Harris, Bill Ryan, and Ted Sjoerdsma, for their assistance and support throughout the conference planning activity.

A huge THANK YOU to Apple Computer for kicking off NECC '87 in fine style by sponsoring the lovely opening reception, and for providing the registration folders.

Finally, the entire Conference Committee wishes to thank all of you who came to visit us in Philadelphia and helped make NECC '87 truly work the way it should.

Frank L. Friedman
NECC '87 Conference Chairman
Temple University

TABLE OF CONTENTS

Spotlight on Delaware Valley Sessions

2 **State and Regional Activities**
Pennsylvania Department of Education Activities
Doris M. Epler
Pennsylvania's Teacher Inservice Programs in Computer-Assisted Instruction
David Dunlop, John Kerrigan, Charles Forsythe

3 **Teacher and Higher Education**
Secret-CSPRO: A Model Program for Re-Training Teachers to Teach Computer Science
Robert M. Aiken, Glen Snelbecker, Arlene Kramer
Computer Technology: A Lens for Teachers
Pamela Freyd, Jessica Kahn

5 **Secondary Uses of Computers 1**
Use of Lap Computers to Improve Students' Writing Ability
Marion H. Rosecky
The Place of the Microcomputer in the Secondary Business Education Curriculum in the Archdiocese of Philadelphia
Sister Rose Lawrence, IHM
UNIX: Multi-User, Multi-Tasking Operating System in the Secondary School
Brother Mark McBride

6 **Secondary Uses of Computers 2**
High Technolgy Track—George Washington High School
Margery Conn, Paul Hampel
The Eagle's Net BBS
Paul Hampel

8 **Elementary Uses of Computers**
The Use of Macintosh Technology in an Elemetary Computer Science Program
Mel Levin
Science Program in an Elementary School
Eileen Adams, Delores Jordan, Patricia Walsh, Alice Wootson
Keyboarding in the Elementary Classroom: Effecting Changes in the High School Business Curriculum
Joanne Romano

10 **Telecommunications**
The Australian/American Connection
Betsy Heslop, Rosemary Hiriak, Greg Butler, Holly Jobe
Telecommunications and the Data Base
Paula Rothman, Paula Halfon, Nancy Lee Bergey

11 **Special Education**
What Do You Get When You Cross a Computer, a Trainable Mentally Retarded Child, and a Gifted Child? Success!
Marilyn Melstein, Joanruth Hirshman

What's New in Technology for Special Educators
Marcia B. Klafter, Brian Fitzgibbons

12 **Integrating Computer Technology into Curriculum**
Building a Solid Foundation for Integrating Computers into the Curriculum
Michael Sinkinson
How to Use Computers to Make Curriculum Integration Easier and More Motivating
Virginia B. Modla, Helen Fox
Development and Integration of Computers into the Curriculum
Mary Ann Coe

14 **Administrative Uses**
Strategic Planning for the Use of Microcomputers in Instructional Programs
John A. Bartley, W. Edward Bureau
Student Records at Your Fingertips with Data Base and Spreadsheet Software
Julianna Csongor
Administrative Uses of the Computer at the Building Level
Richard Luoma, Linda Dean, Paul McEnerney

16 **Industry and Public Libraries**
The Philadelphia School District/Local Industry Cooperative Training Project
John DiLullo
Establishing a Microcomputer Center
Marguerite M. Buck

---

18 **Early Childhood - *Project Session***
Young Children, Robots, and Logo: Cooperative Problem Solving in Three-Dimensional Microworlds
Eric S. Heller, George Forman
Teaching Young Children to Use Computers: Techniques That Work
JoAnn Springsteen, Dene G. Klinzing
The Effect of Developmental Software on Young Children
Susan W. Haugland, Daniel D. Shade

20 **K-8 Writing - *Project Session***
Using Computers to Improve Student Writing
Judy Piper
Reading/Writing/Research: Technology in the Service of Creativity
Evelyn Bender
N. E. W. Visions—Noteworthy Experiences with the New Technology Language Arts Project
Gary A. Carnow

22 **Teaching Algebra With the Support of Technology - *Panel Session (SCS)***
Richard Carter, Chair

23 **Grading Programming Classes - *Panel Session (SIGCSE)***
Harriet G. Taylor, Chair

24 **Teacher Training - *Project Session***
The Impact of the Training Modules for Trainers Project on Training
Elizabeth Uzdavinis Carlson

Preparing Teachers for the Information Age: The University of San Francisco Teacher Credential Program
Janet Graeber, Mary Furlong
Implications of Cognitive Style for Computer Implementation in Middle Schools
Suzanne C. Joseph
Computer Literacy as a Function of Computer Aptitude and Interest Among Preservice Teachers
Ida Malian

26 **BITNET and Beyond: National and International Academic Communications on Your Campus - *Society Session (EDUCOM)***
Michael D'Amore

27 **Software for the Humanities Classroom - *Panel Session (ACH)***
Nancy M. Ide, Chair

Math: High School/College - *Paper Session*

28 Computer-Based Tutor on Graphing Equations
David Trowbridge, Jill Larkin, Carol Scheftic
34 CAI Units on Graphing with Non-Graphics Terminals
Robert J. Lopez
38 Computer Graphics for Differential Equations
Sheldon P. Gordon

45 **Software Licensing: A Positive Response to Real World Problems - *Panel Session***
Mark Friedland, Moderator

46 **Teaching and Using Application Software -*Panel Session***
Arthur Luehrmann, Chair

47 **NSF Elementary Mathematics Materials Grants - *Panel Session (SCS)***
Susan Friel, Chair

48 **Math/Science - *Project Session***
Marble Bags, Balances, and Anti-Marbles: Approaching Algebra Through Concrete Metaphors and Graphic Images
Richard Carter
Assessment of Critical Aspects of Logo Learning: A Collaborative Research Project with Teachers as Observers
Daniel H. Watt, Molly Watt
Effects of CAI Math on Low-Achieving Students: An Eight-Year Longitudinal Study
Clarence DelForge, Milford Clark
Computer Literacy as a Means to Academically Motivate Potential Dropouts
David Sperber

Computer Literacy - *Paper Session*

50 Moving from the Beachhead to Integration with Productivity Tools
David Weaver
53 The Construction of the Computer Competence Tests
Philip M. Linerode

60 **Campus Support Facilities** - ***Project Session***
Project QUEST: Innovative Uses of Microcomputers in Teaching and Research
George H. Culp
Changing Attitudes Towards Computing: A Profile of the Humanities Faculty at the University of Illinois at Urbana-Champaign
Michael J. Jacobson, Martha H. Weller
The Design, Development, and Implementation of a Microcomputer Based Campus Automation System
Timothy J. Foley
Instructor Workstations and Their Classroom Effectiveness
Richard E. Swanson, Gary L. Lorenzen, Thomas E. Gist, Marilyn K. McQuade, Robert G. Fuller

62 **Computers and the Teaching of Literature** - ***Special Session***
Stephen Marcus

63 **A Videotape Survey of Technological Advances in Military Training** - ***Special Session***
J. D. Fletcher

Computer Science: Teaching - ***Paper Session***
64 An Introductory Problem Solving and Design Course
Shirley J. Beil
69 An Educator's Solution to Computer Repair Problems
Richard G. Moll
73 An Interactive Package for Teaching Pascal Pointers and Dynamic Variables
Jeffrey L. Popyack

Models for K-12 Computer Integration - ***Paper Session***
77 Policy and Practice Affecting the Instructional Uses of Computers in Elementary Schools
Robert L. Blomeyer, Jr., George W. Bright
85 The Role of School-Based Computer Coordinators as Change Agents and Staff Developers
Neal Strudler
89 The Drexel University-PRISM Computer Applications in Teaching Program: A Study in University, School District, and Industry Collaboration
Francis M. Betts III, Elizabeth L. Haslam, Fredricka Reisman

95 **Utilizing Videodiscs in Education: Task Force Report from the First National Videodisc Symposium for Education** - ***Panel Session***
Tom Burnett, Chair

96 **The Future of Technology in the Home and School** - ***Panel Session***
Richard Wenn, Chair

97 **Exploring New Directions for Computers in the Curriculum** - ***Special Session***
Henry F. Olds

98 **Apple Classroom of Tomorrow** - ***Panel Session***
Keith Yocam, Chair

Computers in Science - *Paper Session*
99 Tools for Theory Building and Problem Solving
Tim Barclay, Nancy Roberts
103 Conducting a Class in Hypothesis Testing with a Single Computer
Jon B. Latson
108 WCU MicroNet—Update on a Computer Network
Linda J. Perry, Patricia G. Johnson, J. Roger Bacon

Computers in Higher Education - *Paper Session*
114 Teaching a 19th Century Mode of Thinking Through a 20th Century Machine
C. A. Bowers
119 Computers and Basic Economics: Drexel's Experience
Edward Clifford Koziara, Chiou-Shuang Yan
125 Computer-Assisted Instruction and Examination
David M. Sherman

Teacher Training - *Paper Session*
130 Teachers, Computers, and Curriculum Goals: An Inservice Videotape from Down Under
Betz Frederick, Harold Frederick
132 Retraining Teachers in Low Demand Disciplines as Computer Learning Teachers
Gary G. Bitter, Richard Sheets
135 Pennsylvania's Program for Inservice Training for Teachers in Computer-Assisted Instruction
Anne Weiner, Matthew Bodek

137 **Continuing Education Activities of the ACM** - *Panel Session*
Robert M. Aiken, Chair

138 **The Place of Computer Science in the Liberal Arts/Small College Program** - *Panel Session (ASCUE)*
R. Waldo Roth, Chair

Math/Statistics and Expert Systems - *Paper Session*
139 Expert-System Based Instructional Systems
M. C. Lee
143 Computer-Based Interpretation of Students' Statistical Errors: A Preliminary Empirical Analysis of GIDE
Marc M. Sebrechts, Lael J. Schooler, Lafe LaClaire, Elliot Soloway
149 Computer Simulations in Elementary Statistics
Florence S. Gordon

154 **Toward Large-Scale Use of Computers** - *Panel Session*
Alfred Bork, Chair

155 **Bridging the Gap: Strategies for Promoting the Transfer of Learning from Computers to Classrooms** - *Special Session*
Lynne Anderson-Inman

156 **CompuTHINK: Promoting Cognitive Skills** - *Special Session*
Beverly Saylor

157 **Computers in the Schools: The IBM Model Schools Project** - *Panel Session*
Hugh F. Cline, Chair

158 **Integrating Model-Building and Simulation into the High School Curriculum** - *Panel Session (SCS)*
Nancy Roberts, Chair

159 **Pre-Writing Tools for the University Environment** - *Panel Session (ACH)*
Nancy M. Ide, Chair

160 **What are Colleges Doing about Teacher Training?** - *Special Session*
Dave Moursund, Charlotte Scherer, Gary G. Bitter, Larry Hannah

162 **Spotlight on Industry** - *Special Session*
Learning to Exploit the Information Center
Harry Albright
Videodisc Use in Management Development: A Discovery Learning Case
Verl Dennis

Computer Science: Teaching Programming - *Paper Session*
163 Cognitive Barriers to Initial Programming Instruction: A Survey of Research
William Wresch
168 The Graphical Representation of Programming: Research into Alternatives for Learners
Nancy Cunniff, Robert P. Taylor
175 The Validity of Programming Ability Tests
David M. Olson

182 **Computer Adaptive Testing** - *Panel Session (AERA)*
Thomas H. Saterfiel, Chair

183 **Preview Centers and Publishers: A Cooperative Venture** - *Panel Session (ICCE)*
Vicki Smith Bigham, Chair

Early Childhood - *Paper Session*
184 Kindergarteners Discover Robots
Elizabeth Anne Viau
190 The Relationship Between Home Microcomputer Ownership and Development of Kindergarten Students
Thomas E. Franklin, Ronald Zoromski
200 Strategies for Sequential Phonics Instruction Using Computer Software
Linda Mensing, Anne Meyer

204 **Social Studies/Applications** - *Project Session*
Primary Students' Uses of Data Bases as Tools for Lifelong Learning
C. Jane Hydrick, Kay Gore
Study and Information Skills—Interactive Video Style
Joanne Troutner
Appropriate Applications of Educational Computing for Elementary School Science and Health
James D. Ellis, Rodger W. Bybee
Camp at the End of the Universe
Paula J. Panto, Mark Nelson

206 **Applications** - ***Project Session***

Using *Lotus* 1-2-3 to Teach Information Processing Basics

Thomas R. Harris

Replacing Computer Literacy

Joseph R. Meadors

Cryptographers, Thunderstorms, and Computer Games: A Computer-Based Approach to Functions and Graphing Linear Equations

Judith Montgomery

Building a Communications Network

Francis M. Betts III, Sarah Poncz

English/Language Arts: College - ***Paper Session***

208 Computer Aids for the Teaching of Critical Reading in the Liberal Arts Curriculum

Joseph Orndorff

213 Comparison of Computer-Based and Traditional Practice Exercises for Development of Content Area Vocabulary at the College Level

Ernest Balajthy, Linda Bacon, Patricia Hasby

217 Desktop Publishing: An Introduction

Frank Kulesa

227 **Perspectives on Logo Teacher Training** - ***Panel Session (ICCE and Logo Exchange)***

Doug Clements, Chair

228 **Materials Design** - ***Project Session***

Software Evaluation: How Do Teacher Reviews of Software Compare to Classroom Results?

Ronald D. Owston, Herbert H. Wideman, Curt Dudley-Marling

Using Interactive Videodisc in an Introductory Computer Class

Nancy P. Houston

CAI Programming and Student Learning Styles: Matches and Mismatches

Debra K. Stein, John Knauer

229 **UNIVAC: The Early Years** - ***Special Session***

Joanne W. Iverson, Stephen Longo, Raymond Frankson, Howard Masters, Rick Cloughly, Paul Chinitz, Cedrick Lee, Saul Brownstein, Betty Jean Bartik, Jerry Smoliar, Al Tonik, Rich Berryman, Jerry Feldscher, Laura Cousins, Jim McGarvey, Fred Frankel

230 **Computer-Assisted Language Instruction** - ***Panel Session (ACH)***

Randall Jones, Chair

231 **Research about Effectiveness: Self-Deception and Reality in Surveys, Program Evaluation, and Field Experiments** - ***Special Session***

Henry Jay Becker

232 **Training Executives Worldwide** - ***Special Session***

Sandra Morris, Chair

233 **Special Needs** - ***Project Session***

Development of an Individualized Computer Curriculum for Gifted/Talented American Indian Students

Jo Lynn Autry Digranes

CMES and the Hearing Impaired—New Ways of Sharing Computer Technology
Phil Mackall
Outfitting the Expedition: Equipping the Special Education Student with Lifetime Skills
Leon Foster
Computers in Correctional Education: Report of a 1986 National Survey and Recommendations
Antonia Stone

235 **Videodisc Instruction in Math and Science - *Special Session***
Douglas Carnine

236 **University Computer Science - *Project Session***
An Investigation of Prior Sources of Difficulties in Learning University Computer Science
Victoria Dixon
Sex Differences in Attitudes Toward Computers
Elsa V. Parsegian, Howard W. Pullman
Problem Solving Emphasis in a Computer Science Curriculum
Richard A. Alo'

237 **Resources for Secondary Computer Science Teachers - *Panel Session***
Vicki Smith Bigham, Chair

Development and Evaluation of Software - *Paper Session*

238 Use of Microcomputer Software in the Development of Multiple-Choice Reading Tests for Elementary and Secondary Public School Students
Elisabeth G. Flach
248 A Customized Authoring System for Rapidly Programming Basic Competencies
K. Wayne Goode, Bernard J. Schroer
253 Instructional Feedback and Educational Software
Michael Land

258 **Preparation for College Computer Science - *Panel Session (SIGCSE)***
Harriet G. Taylor, Chair

Computers in Library and Language - *Paper Session*

259 CD-ROM, CD-I, and the "Information Revolution"
Francis A. Harvey
268 Children's Use of a CD-ROM Encyclopedia: A Look at Electronic Information-Seeking Strategy
Gary Marchionini, Diane Patrick, Jerry Teague
273 Planning for Computer Enhanced Instruction
Gertrude W. Abramson

277 **EDUCOM Software Initiative—Update and Opportunities; Emphasis: The EDUCOM Code - *Society Session (EDUCOM)***
Steven W. Gilbert, Diane Balestri

278 **Spotlight on Business and Industry - *Special Session***
Seeding Expert System Technology in Industrial Settings
Lynne M. Davis

Computer-Based Learning in Corporations: Challenges and Opportunities for Creative Lifelong Learning
Ted Kahn

279 **Administrative Use** - ***Project Session***
Computerized Adaptive Testing
Lucky J. Abernathy
Manhattanville College Expert Academic Advisor—Preliminary Report
Gerard Kiernan, Arnold Koltun, George Psihountas, Edward N. Schwartz
The School Board: An Electronic Bulletin Board for Educators
Sandra V. Turner
The New Jersey Science Supervisors' Electronic Conference
Mark O'Shea, Howard Kimmel

281 **Technology in the Foreign Language Curriculum: A Resource for Teachers** - ***Special Session***
John A. Vaille

282 **ICCE Certification Recommendations for Computer-Using Educators** - ***Panel Session (ICCE)***
Tony Jongejan, Chair

283 **Applications** - ***Project Session***
A Computing Survival Skills Course for Liberal Arts Colleges
Robert J. Aiken, Douglas L. Peterson, Ann E. Commito
Implementation of Microcomputers in Mathematics Courses at The United States Military Academy
David C. Arney, Anthony J. Manganiello, William P. Rue
A Survey of Microcomputing Courses for Health Educators
Ernest A. Randolfi
Transforming On-Line Computer-Assisted Instructional Displays: An Investigation of Two Audience Analysis Programs
Ann Hill Duin

285 **Moving Toward Curricular Integration: Merging Sound Instructional Practices with Appropriate and Effective Computer Software** - ***Special Session***
Lynn M. Lieberman

Ethics, Equity, and Special Needs - ***Paper Session***
286 Computer Implications and Ethics: A First but Last Priority
George G. Bear
291 Enhancing Minority Student Educational Opportunities through Instructional Computing: A Report on Project MICRO
Carol Edwards, Jacqueline Jordan Irvine
298 Special Student-Computer Connection
Beverly Cunningham, Lee Cunningham

302 **High School Computer Science** - ***Project Session***
Robotics in the Classroom: A Source Book and Data Base
Kathleen Zurhellen
The Transfer of Problem Solving Skills from Programming in a First and Second Language
Diane McGrath

Math/Special Needs - *Paper Session*

303 A Model CAI Program for Applied Mathematics
Mary Garrett
309 Mathematics and Problem Solving: Computer Based Strategies for Instruction
Bethany N. Roditi, Anne Meyer

318 **Computer-Based Learning Environments** - *Panel Session*
Gertrude W. Abramson, Chair

319 **Evaluation of District Instructional Computer Programs** - *Panel Session*
Karen Billings, Chair

Special Session

320 Data Bases in the Classroom: Beneficial for All Students?
Don Rawitsch
321 Data Bases and Spreadsheets in History and Social Studies Inquiry
Larry Hannah

Special Session

322 Spreadsheets in the Mathematics Curriculum: 6-12
Tony Jongejan
323 Impact of Computers on the Mathematics Curriculum
Judi Mathis

324 **Junior/Community College** - *Project Session*
Courseware Authoring in All Departments at Freed-Hardeman College
J. D. Thomas
Computers and the Small Media Center
Dale Pugsley
Instructional Computing in Washington's Community Colleges
Margaret "Kit" Ellis
Computers, Disabled Students, and the Community College: Adapted Computer Studies for Disabled Students
Jacqualine Couch

Methods for Teaching Computer Science - *Paper Session*

326 A Computing Methods Course: Content, Organization, and Activities
Lajeane G. Thomas
331 The Development and Piloting of a Methods Course for High School Computer Science Teachers
Charles S. White, David C. Rine
335 Logic Programming as a Computing Environment for Documenting Teacher Decision-Making
Anne Shelly, Ernest Sibert

341 **Expanding Computer Learning for Disadvantaged: Quality and Quantity** - *Panel Session (ICCE)*
Antonia Stone, Chair

Logo - *Paper Session*

342 Bugs in the Workshop: Using Logo to Enrich Language Experience
Pamela Freyd

348 Logo, Lists, and Treasure Hunting: The Use of Multiple Concrete Models in the Teaching of Computing
Karen Swan, Robert P. Taylor, John B. Black
353 Empowering the Teacher—Creating a Logo Culture in Less than Ideal Situations
Gary S. Stager

Health/English - *Paper Session*
360 Computer-Assisted Health Promotion: Current Developments
Ernest A. Randolfi, David Bacharach
366 Computer-Assisted Programs for Assessing Cardiovascular Fitness
David Bacharach, Ernest Randolfi
370 Automated Plagiarism Detection: A Simple Approach
James L. Wolfe

376 **Identification and Selection of Software - *Panel Session***
Vicki Smith Bigham, Chair

377 **'Shifting Sands' in the Undergraduate Computer Science Curriculum - *Panel Session***
Allen B. Tucker, Moderator

Special Session
378 Keyboard Connections: Using FrEdWriter in the Classroom
Greg Butler
378 *FrEdWriter, FrEdSender*, and Telephones: A Practical Approach to Networking Activities
Al Rogers

379 **Applied Math, Field Science, and Piaget - *Special Session***
M. Heidi Imhof

Special Session
380 Computers in a High School Science/Math/Computer Science Magnet
Michael Haney, Lola Piper
381 On-Line Searching for a Biology Research Project
Erica K. Lodish, Gloria A. Seelman

382 **Applications - *Project Session***
Developing Student Computer Literacy with Lap-top Computers
David Miles, Thomas Flynn
Extending the Classroom Environment with Comunications Tools
James TenEyck
Computers Across the Curriculum: The First Year
Nick Gordon, Bob Latzer, Jill Lewis
Computer Instruction Within the Schools of Business Administration
Ronald D. Schwartz

384 **K-12 Integration - *Project Session***
Effective Inservice for Computer-Integrated Instruction
Dave Moursund
The Changing Roles of the School District Computer Coordinator
Jean Dames, Ann Dana, Michael P. Warner
Using Technology to Improve the Learning of Problem Solving, Science, and Writing Skills
Gerard J. Berry, Francine L. Gallagher, James D. Phifer, Ronald E. Zirkle

386 **Integrating Educational Software into Home and School Learning** - *Special Session*
P. Kenneth Komoski, Wells Hively

387 **A Cognitively Motivated Approach to Teaching Computer Programming** - *Panel Session*
Elliot Soloway, Chair

388 **Not to be Taken Out of Context: Research on Computers in the Classroom** - *Panel Session*
Betty Collis, Chair

389 **National Programs to Improve Education Through Technology** - *Panel Session (ICCE)*
Andrew R. Molnar, Chair

PROCEEDINGS

OF THE

NATIONAL EDUCATIONAL COMPUTING CONFERENCE

JUNE 24 - 26, 1987

PHILADELPHIA, PENNSYLVANIA

STATE AND REGIONAL ACTIVITIES

Dr. Doris M. Epler
Dr. David Dunlop
Dr. John Kerrigan
Mr. Charles G. Forsythe

ABSTRACT: Pennsylvania Department of Education Activities

Doris M. Epler, PDE, 333 Market Street, Harrisburg, PA 17126-0333

The Pennsylvania Department of Education has indeed been a leader in using microcomputers to enhance education opportunities for its students, teachers and administrators.

We were the first state in the nation to integrate online searching into the secondary school library media curriculum. The LIN-TELL (Linking Information Needs: Technology, Education, Libraries) network, consisting of 120 members, was established to provide schools with access to a commercial database, BRS (Bibliographic Retrieval Services). Support services included training for school librarians in online searching, user meetings, newsletters, an electronic message system and the development of PENNSYLVANIA ONLINE, which is a curriculum guide to assist librarians to integrate online searching into the school library media curriculum. In addition, annual contests are conducted to recognize the Outstanding Student Searcher of the Year.

Pennsylvania was also able to achieve another national first when it created the first union library catalog on compact laser disc. This database holds information about 153 schools, public and and academic libraries. The 653,000 unique titles which were derived from more than 2.7 million items, can be searched by subject, title, author or any word in just a few seconds. The database itself can be defaulted from statewide, to regional, to consortium or to individual library. In just a few weeks, the libraries represented on this disc have reported a drastic increase in their circulation as well as inter-library loan requests.

Another exciting Pennsylvania project is "Tele-learning". The goal of this project is to provide cost-effective education to groups of students who are currently excluded from many educational programs because they exist in insufficient numbers. By linking microcom-puters and telephones together, students now have access to these educational programs.

Pennsylvania is indeed a leader in using technology to enhance the teaching/learning process.

ABSTRACT: Pennsylvania's Teacher Inservice Programs in Computer-Assisted Instruction

David Dunlop, University of Pittsburgh at Johnstown, Library, Room 34, Johnstown, PA 15904
John Kerrigan, West Chester University, West Chester, PA 19380
Charles Forsythe, Montgomery County Intermediate Unit, Montgomery Avenue and Paper Mill Road, Erdenheim, PA 19118

This presentation will focus on programs in the state of Pennsylvania that are concerned with training classroom teachers in computer-assisted instruction. Dr. Dunlop will present the Information Technology Education for the Commonwealth programs, operations and documented success. He will also present the ITEC courses and their developed and describe the operations and programs at the Regional Computer Resource Centers with emphasis on the center at West Chester University.

Mr. Forsythe will describe how the Intermediate Units in Pennsylvania are set up to meet the needs of the districts. He will discuss the microcomputer activities of some of the Intermediate Units in the Delaware Valley. He will emphasize the close working relationship between the districts, Intermediate Units and Regional Computer Resource Centers. He will discuss the goals of the I.U. programs, activities, special projects, school consulting, etc.

The session will conclude with the panel members responding to questions from the audience.

TEACHER AND HIGHER EDUCATION

Dr. Robert M. Aiken
Dr. Glenn Snelbecker
Ms. Arlene Kramer
Ms. Pamela Freyd
Ms. Jessica Kahn

ABSTRACT: Secret-CSPRO: A Model Program for Re-Training Teachers to Teach Computer Science

Dr. Robert M. Aiken, Dr. Glen Snelbecker,
Temple University,
Phila., PA 19122
Mrs. Arlene Kramer, Philadelphia
S.D., Phila., PA 19103

The primary goal of this project is to discover how effective a model, developed by Drs. Aiken and Snelbecker, is in re-training junior and senior high school teachers with a wide range of backgrounds to teach Computer Science in senior high schools. This model is based on one published by an ACM Committee; "Computer Science for High Schools: Course Content." This two-year project (funded by the National Science Foundation) will provide three graduate level courses in Computer Science and one Instructional Design Practicum for two groups of teachers of 25 teachers each. The following topics will be discussed:

1. Summary of the project's goals and an indication of how well these goals have been met (10-15 minutes).

2. Overview of the role of the Philadelphia School District's Computer Science Technology group, including a discussion of the Exam they give as one aspect of certifying their Computer Science Teachers (10-15 minutes).

3. A collage of experiences of the staff (15 minutes).

4. Panel discussion featuring several of the teacher/students discussing their experiences (15-20 minutes).

The major objective of this session will be to share with others the preliminary results of this nationally funded Computer Science Retraining Program. With school systems throughout the United States facing a severe shortage of Computer Science teachers at the pre-college level, we believe that our experiences may be helpful.

ABSTRACT: Computer Technology: A Lens for Teachers

Pamela Freyd, Jessica Kahn
School of Education, University
of Pennsylvania, Philadelphia,
PA 19104

The Graduate School of Education at the University of Pennsylvania offers a Master's Degree in Computer Education. Some students combined a Master's Degree and certification program in elementary or secondary education, conjunction with the Teacher Training Faculty. At the doctoral level, students integrate computer courses with their interests and specializations within the Educational Leadership program.

The goals of the Computer Education program at the Graduate School of Education are the following:

To help teachers build a proficiency in using computer technology Toward this goal, programming courses are offered in LOGO, Basic and Pascal. Students learn how to use word processing, databases and spreadsheets. The political, economic, social and ethical issues surrounding computer use are explored.

To examine the research and expectations surrounding computer use in schools Our courses critique the growing body of research on computer use in schools, and challenge the student to create research questions and designs to address those questions.

To prepare teachers to integrate Computer technology into the existing curriculum Creation of lesson plans and units which support and extend the existing curriculum is a vital part of our program. These lesson plans and units are written by the teachers for use in their particular instructional settings. We also address the question of software design with an emphasis on educational merit rather than technical proficiency.

To help teachers use computer technology as a lens through which to evaluate current teaching/learning practices As educational technology is introduced in the schools, an opportunity arises for re-assessing how children and teachers develop educational values and the ways in which teachers might realize those values in practice. Use of any technology, any innovation must be informed by a philosophical stance which makes the educational process an integrated experience.

SECONDARY USES OF COMPUTERS
Part 1

Dr. Marion H. Rosecky
Sister Rose Lawrence, IHM
Brother Mark McBride

ABSTRACT: Use of Lap Computers to Improve Students' Writing Ability

Dr. Marion H. Rosecky, Upper Dublin
High School, 800 Loch Alsh Ave.
Fort Washington, PA 19034

The Business and English Departments of the Upper Dublin High School cooperated in writing a proposal to use Lap Computers to improve students' writing ability. This proposal was approved and funded by Radio Shack for the 1986-87 school year.

The questions to be answered are:

Will students improve their writing ability using the Lap Computer? Will students have more positive attitudes toward writing by using the Lap Computer? Will students be more willing to revise their drafts when using the Lap Computer? Will teachers find the teaching of writing to be more effective by using the Lap Computer? How? Why?
Can we apply this instructional design to other English classes in this high school? In other schools?

The program will be piloted in one Journalism class, one twelfth-grade English honors class, and one eleventh-grade English class. Classes similar in type and level will be assessed as control groups.

Other features of this program include: a telecommunications component, access to printers, storage of drafts on discs, access to an "electronic bulletin board" at the regional Intermediate Unit.

ABSTRACT: The Place of the Microcomputer in the Secondary Business Education Curriculum in the Archdiocese of Philadelphia

Sister Rose Lawrence, IHM, Business
Education Curriculum
Chairperson, Archdiocese
of Philadelphia
Archbishop Wood H.S., Girls
Division, 655 York Road,
Warminster, PA 18974

Highlights on the current status of microcomputers throughout the Archdiocesan Secondary School System and plans for the incorpor-ation of their use in the upcoming years.

It is our vision that all Catholic school students should have a general knowledge of the word processor and other computer applications for use in their every day life. Business students should have an indepth study in the same topics as well as communication and electronic spreadsheeting.

ABSTRACT: UNIX: Multi-User, Multi-Tasking Operating System in the Secondary School

Brother Mark McBride, T.O.R., Bishop
Egan High School, 611 Wistar Rd,
Fairless Hills, PA 19030

The three areas that this seminar will cover are:

A. Operating System
B. School Environment
C. Future

I. Operating System:
  a) Components: Kernel, File System and the Shell.
  b) Define and Discuss Multi-User, Multi-Taking and interactive system.
  c) Networking hardware and information.

II. School Environment
  a) Philosophy of the School relative to computers.
  b) Interdisciplinary usage.
  c) Word and text processing.
  d) Database System.
  e) Integration of Services.
  f) Languages.
  g) On-line capability.
  h) Security.

III. Future
  a) Growth of databases.
  b) Expansion.

Information relative to all these topics will be typed and given to the participants.

SECONDARY USES OF COMPUTERS
Part 2

Margery Conn
Paul Hampel

ABSTRACT: High Technology Track - George Washington High School

Margery Conn, Paul Hampel,
George Washington H.S.
Bustleton Avenue & Veree Road,
Philadelphia, PA 19116

Washington High School's High Technology Track began in September, 1985, for selected students who have academic ability and an interest in computers. The main goal of the program is to produce students who will be highly proficient in using computer applications and who can apply their learning to assignments in other subject areas.

Currently we have 84 ninth grade students who, in addition to a full academic program, take a full year course in Software Applications on the Macintosh computer and a half year course in Telecommunications. They learn the fundamentals of using a computer to communicate and how to use on-line data bases. The plans for the future include another full year course in Advanced Macintosh software covering presentation graphics, desktop publishing, and whatever software is current and useful in 1988.

Our principal, Dr. Carol A. Wacker, has been the driving force behind this program, making possible the facilities and supplies needed. Our Macintosh lab has 32 Macintosh 512 computers and 14 printers. There are three other computer labs in the building as well as public access computers available to students in the library and Mentally Gifted Room. Modems are available in two locations.

Instruction is directed at teaching the mechanics of applications and at illustrating meaningful uses of the applications to the students' needs. The students are enthusiastic to their academic work. Most all of them turn in word processed reports for English, History, and Science. They love the footnoting capabilities of Microsoft's Word application. MacPaint pictures have been used for book covers, pasted entries in written reports, political cartoons, maps, and art contests. Histograms, line graphs, and pie graphs have been used in science lab reports and for graphic presentations by several of our science competition award winners. Some students have put spreadsheet learning to use in the national Stock Market Game. Telecommunications has enhanced the students' ability to obtain current information and to retrieve bibliographic references.

Besides discovering the ways the computer can be applied to their work, the students are quick to appreciate the valuable time the computer saves them. They are becoming more efficient and more aware of the world around them.

ABSTRACT: The Eagle's Net BBS

Paul Hampel, George Washington H.S.
Bustleton Avenue & Veree Road,
Philadelphia, PA 19116

The Eagle's Net is a project designed by the students of George Washington High School's Mentally Gifted Program to facilitate the use of telecommunications equipment and computers in the community and in the school. Our goal is to prepare our students for a computerized future and to increase their communications skills.

Through the use of computers using modems, users interact with an electric bulletin board system via the telephone. The electronic bulletin board concept is similar to a public cork board, where people pin up messages for others to read. The system functions in both local (at school) and remote (from outside of school) modes. During school hours, students can work in the local mode at the host computer. They can interact with it in the same way from outside the school in remote mode.

Information may be exchanged on many topics and in many formats. The system features private mail, registered mail, feature files, uploading and downloading, as well as 49 public bulletin boards. The public bulletin boards cover, among other things, such topics as Debate/Politics, Computer, Dreams, Literary Works, G.W.H.S. News, Science Fiction, Sports, Movies, Music and Videos, Math and Science Problem Solving, French Only, and several Special Interest

Groups (SIGS) for various brands of computers. The system operators (sysops) decide on the topics to be covered on each board. Sysops are chosen from students in the mentally gifted program at Washington HS and serve for one year.

The Eagle's Net currently operates on a 64K Apple IIe computer utilizing a 10 megabyte Sider hard drive. Communication is provided by a Hayes Micromodem IIe. A Proclock time card allows the system to keep track of dates and time. It also measures how long each user is on-line.

Currently the system is averaging 60 calls per day. Callers are mainly from the Philadelphia area, but users to call from the tri-state area surrounding George Washington H.S. Because one of our goals is increased communications, users are not limited to George Washington students. In fact, public access is encouraged. A recent survey indicated that users range from elementary school through university level (many adults also find the BBS a good place to call). The Eagle's Net welcomes all calls.

One of our educational goals is to share the experience of cross cultural communications with our students. We also believe that this new vehicle for information exchange is vital to the understanding of the world around us. To aid this goal, users are permitted the use of pseudonyms. This practice has encouraged honest and frank communications. Many users engage in daily dialog without ever knowing the actual identity of the person they are interacting with. This practice has encouraged many women to use the BBS without fearing the peer pressure of "invading" the usually male dominated computer world. Sex, race, religion, socio-economic status, and physical attributes are all invisible to the users of telecommunication bulletin boards. The Eagle's Net has allowed younger students to communicate with older students, teachers to consult students for information, college students to share information with high school students, all under the protective anonymity of pseudonyms. Teenagers, as well as adults, suffer under the burdens of fearing to communicate. The BBS has gone a long way in breaking down that fear.

Many bulletin boards have gotten a bad reputation for encouraging software piracy, pornography, or other antisocial behaviors. Our sysops check the boards everyday for messages that are not appropriate for public viewing or messages that don't fit in with the philosophy of the BBS.

The system operates 24 hours per day and 7 days a week. From September 1 through June 26, System Operators are available between the hours of 7:00 AM and 2:00 PM (E.S.T.). New users can expect fill validation within twenty-four hours.

The Eagle's Net can be reached at (215) 969-3384. Give us a call!

ELEMENTARY USES OF COMPUTERS

Mel Levin
Eileen Adams
Delores Jordan
Patricia Walsh
Alice Wootson
Joanne Romano

ABSTRACT: The Use of Macintosh Technology in an Elementary Computer Science Program

Mel Levin, Prince Hall Schl, Gratz & Godfrey Ave, Phila. PA 19141

The invention of the high-speed electronic computer has been heralded as the 3rd Information Revolution, preceded by written language and the printed book or as the ultimate power machine, capable of governing the world. Regardless of how we may view the computer, it is here and here to stay. It is reasonable that the potential leaders and decision makers that are entrusted to our training, would profit from an introduction to the newest technology available.

How, we may ask, is one of the most controversial technological advances of modern time, applicable to the elementary curriculum. At Prince Hall we see the computer science program as all encompassing. Our philosophy is that computer science program focuses on four main topic components to form a central core curriculum for our young students: problem solving, computer literacy and technology, tool applications and affective education.

Prince Hall was given a grant by Apple Inc. in September 1986 of a complete Macintosh lab consisting of:

* 18 - 512K enhanced Macintosh computers * 18 -External 800k disk drives
* 6 - Image Writer II printers * 1 - Laser Printer
* Apple Talk network
* Software packages including MacWrite, MacPaint, and MacDraw

The project was to determine if elementary school children could learn and assimilate Macintosh technology.

Our lab now houses 36 computer systems consisting of 18 Apple IIe systems (which were already in place) and 18 Macintosh systems.

Each if the classes (except the 3rd graders) come to the lab twice a week. One period a week the student works with an Apple II using LOG in a problem-solving environment. During the second period of the week the student is on the Mac, creating their own "worlds of desktop publishing".

We use the Logo Programming Language as a very successful vehicle for teaching problem solving skills. Students are challenged with modular, logical, and liner thinking tasks. The nature of the Logo Programming Language promotes experimentation, self-discovery, and positive reinforcement.

Our literacy component includes such areas as: history of computers, parts of a computer, how computers work, and computer related careers. Students do research and write reports. We have been fortunate to visually experience the rapid change in computer technology and to appreciate the relative ease with which varying tasks can be performed. Out students have seen and used in a relatively short period of time 3 Apple products: Apple II Plus, Apple IIe, Macintosh.

The tool applications of the computer have been varied and widely adapted throughout the school building. Primary grade teachers have been using the Macintosh to print the stories and poems their young students have written. They've used a variety of creative writing activities to develop the stories. The idea of having their work "published" was an added incentive to the children to write. They were thrilled when they saw their work in print. They will make an individual book with all of their stories and poems to take home at the end of the school year.

At Prince Hall, reading tests are given at the end of each unit in our Ginn Readers. By using the Ginn Computer Test Scoring Program, Mrs. Carol Marcune with an Apple computer is able to give teachers detailed test results and in addition, print parent letters to keep parents informed of their child's progress.

The computer science reports are also generated by computer. Each student's record is kept on file and can be accessed at a moment's notice. Each report period a summary of every student's progress is distributed to the student.

The Parent Co-op Nursery Program has benefited from the Computer Room in many ways. The children have been exposed to computer terminology. They know how to manipulate the mouse. They have used the laser printer. They have been using programs to reinforce shapes and numbers and letters. They have been taught in an individualized program developed by the sixth grade M.G. Students. Their parents have also been exposed to computers and are surprised by their own child's expertise in handling equipment.

Using our Mac computer and a modem, Prince Hall students have been exploring the world of telecommunications. We have been going on-line with fellow students at George Washington High School. Our students have learned the power and quickness of transferring files electronically. They have experienced the marvel of "electronic mailmen".

Our school newspaper has now become "computerized". All club reporters were introduced to the MacWrite Word Processing Program and MacPaint Graphics Proram so that this issue became our first Mac-issue. Students typed, revised and edited their articles using the computer.

The initial results have far exceeded our expectations. Students are assimilating Mac technology at an incredible rate. We are extremely proud of the work they have produced thus far. Although we have just "scratched the surface", it seems that elementary school children can successfully incorporate "state of the art" technology in a computer-based educational experience.

ABSTRACT: Science Program in an Elementary School

Eileen Adams, Delores Jordan, Patricia Walsh, Alice Wootson,
C.W. Henry School, Carpenter Lane at Greene Street,
Philadelphia, PA 19119

The presenters, teachers with expertise in such subject areas as language arts, reading and mathematics, will describe their team approach to the development of a computer science program at the elementary level. This program is in its fourth year of operation at the Charles W. Henry School, School District of Philadelphia. It is replicable in whole or part by similar organizational models. Highlights of the presentation will include the following:

1. Rationale for the team approach.

2. Pilot phase which included formulation of written plan containing license requirements for programming skills, word processing and awareness objectives.

3. Facets of staff training: pre-pilot phase training of team members; on-site staff development and graduate credit courses (Beaver College) open to Henry staff and other interested educators; on-going Henry School staff development in use of card reader, math and reading management systems, Grade Book, Grade Manager; MECC software evaluation sessions by entire Henry staff using School District of Philadelphia evaluation forms, etc.

4. Expansion of program to include a second computer lab and the creation of first elementary telecommunication bulletin board system for the School District of Philadelphia (sub-board to PRISM HTT).

5. Summary of advantages of the Team Approach.

ABSTRACT: Keyboarding in the Elementary Classroom: Effecting Changes in the High School Business Curriculum

Joanne Romano, St. Dennis School,
200 Eagle Rd, Havertown, PA 19083

During the Fall of the school year 1986-87, representatives of the Elementary/ Secondary Computer Committee of the Archdiocese of Philadelphia met with representatives of the High School Business Education Committee to discuss the important role of the teaching of keyboarding skills to the elementary school children in the Archdiocese. As a result of this meeting, the committees projected that the teaching of keyboarding skills and other tools (word processing, spreadsheets, databases and LOGO) will necessitate changes in the high school business curriculum. Some classes now taught in high school may possibly be eliminated.

This talk will focus on the suggestions and projections made by both committees and on the surveys sent to the elementary schools regarding the teaching of keyboarding skills and tools and on the joint venture of the committees to strengthen the curriculum from K-12 in this computer age. Practical tips on how to implement changes will be discussed.

TELECOMMUNICATIONS

Betsy Heslop
Rosemary Hiriak
Greg Butler
Holly Jobe
Paula Rothman
Paula Richards
Nancy Lee Bergey

ABSTRACT: The Australian/American Connection

Betsy Heslop, Rosemary Hiriak, West Pottsgrove Elementary School, Grosstown Rd, Pottstown, PA 19464
Greg Butler, Miller Education Center, Miller, New South Wales, AUSTRALIA
Holly Jobe, Montgomery County Intermediate Unit, Montgomery Avenue & Paper Mill Road, Erdenheim, PA 19118

This presentation will follow the conception and development of a telecommunications project linking two groups of fourth and fifth grade students on opposite sides of the world.

The presenters will include the microcomputer specialist from the Montgomery County IU who coordinated the project, as well as the participating teachers (from both the US and Australia) and several of the students from the West Pottsgrove Elementary School in Stowe, Pennsylvania. Each presenter will share the project from his or her unique point-of-view.

The goal of the county resource person, to help teachers and students utilize the technology of telecommunications as a tool within the elementary school curriculum, will be shared and contrasted with the goals of the classroom teachers whose primary concerns lie within the classroom curriculum areas. As the project ideas are developed and expanded upon, it will become clear that the technology of telecommunications has indeed become a tool through which reading, writing, language arts, and social studies skills are meshed in a common project.

The student correspondence shows evidence of the development of skills in all of the above areas. What is more apparent than those "school" skills is the excitement of being able to reach out and discover first-hand that students on the other side of the world are real kids who have Cabbage-Patch dolls and transformers, who study history and math in school, who go on school camping trips and who have brothers and sisters and dogs and cats. It is even more exciting to be able to share differences--in holidays, heritage, seasons and foods.

Watching fifth grade students casually access an international electronic mail system as easily as they telephone a friend down the street--is it just a novelty--or is it the door opening onto a world of better communication and understanding?

ABSTRACT: Telecommunications and the Data Base

Paula Rothman, Paula Halfon, Nancy Lee Bergey, Solomon Schechter Day School of Philadelphia, Old Lancaster Rd & Highland Ave, Bala Cynwyd, PA 19004

This project integrates the use of the modem and data base in the sixth grade science curriculum between the two branches of the Solomon Schechter Day School of Phila. The main goals of the project are to show the modem as a tool for electronic data base searching, sharing information, and the designing of a data base to be uploaded and downloaded between branches in order to foster feedback between the students at each school.

This presentation will outline the steps taken to set up the modem in the schools as well as implementing its use. Information on how the children were instructed to design their data base and how they learned to use the modem will be given. An overall evaluation of the project will be shared.

SPECIAL EDUCATION

Marilyn Melstein
Joanruth Hirshman
Marcia B. Klafter
Brian Fitzgibbons

ABSTRACT: What Do You Get When You Cross a Computer, a Trainable Mentally Retarded Child and a Gifted Child?: Success!

Marilyn Melstein, B.Rush Middle School, Knights & Fairdale Rds, Philadelphia, PA 19154
Joanruth Hirshman, Comegys School, 51st & Greenway, Philadelphia, PA 19143

A computer project started in the Spring of 1984 involving 45 mentally gifted students and 11 trainable mentally retarded students. The project originated as an effective means of meeting the unique communication needs of one non-verbal adolescent male with cerebral palsy, and grew to include all of the students in the class. A computer and voice synthesizer were utilized by the MG students to develop original computer programs for the TMR students for communication and skill reinforcement. The union of these two exceptionalities has provided a myriad of benefits for both groups. The most important benefit being the observable respect for the value of each individual by all the participants.

ABSTRACT: What's New in Technology For Special Educators

Marcia B. Klafter, Brian Fitzgibbons
RRC/PRISE, 200 Anderson Road,
King of Prussia, PA 19406

Many new computer peripherals and software have become available over the last year that offer direct and innovative applications for special education. This session will demonstrate and discuss many of these new Apple based materials. Multi-modality word-processors and other software that utilize new input/output approaches such as the TouchWindow and the Power Pad will be examined for their applicability to different special populations. The features of the Apple IIGS and recently developed software and hardware that take advantage of the machine will also be considered in light of needs of the exceptional student. The presenters will also discuss how some of the newer software tools have been modified and expanded so that youngsters with sensory, learning or physical disabilities can enter the mainstream of computer use.

Computer training and technical assistance for special educators in Pennsylvania is provided through a unique network of three regional centers. This session will outline the variety of services and the model for service delivery that enables anyone working in an educational program for the handicapped to get the support needed for effective computer use. A comprehensive, computerized IEP package, Pennstar, developed by the State but available at minimal cost to any interested school district will be demonstrated.

INTEGRATING COMPUTER TECHNOLOGY INTO CURRICULUM

Michael Sinkinson
Virginia B. Modla
Helen Fox
May Ann Coe

ABSTRACT: Building a Solid Foundation for Integrating Computers into the Curriculum

Michael Sinkinson, Methacton S.D.,
Fairview Village, PA 19403

Goals: To focus on prerequisites for developing a meaningful "computers in the curriculum" program.

Prerequisite #1: Political support; Board, Administration and Community. Ways to get people on your side.

Prerequisite #2: Solid "change management" practices. How to set the environment for computers. The importance of long-range planning and frequent updates of plans.

Prerequisite #3: Teacher support. Things to do (and not do) in getting teacher support for using computers.

Prerequisite #4: Teacher knowledge. Various in-service and other educational models; advantages and disadvantages. Importance of software and hardware familiarity prior to classroom use.

Prerequisite #5: Identifying educational needs. Why it's important to focus on curriculum, not computers. How much of the student population needs what kind of use of the computer?

Prerequisite #6: Having someone oversee the project. Why this is not a part-time job. The qualities and qualifications someone needs to do this job.

ABSTRACT: How to Use Computers to Make Curriculum Integration Easier and More Motivating

Virginia B. Modla, Cheltenham Twp
Schools, Ashbourne Road and
Washington Lane,
Elkins Park, PA 19117
Helen Fox, Elkins Park Middle
School, New Second Street
and Tookany Drive,
Elkins Park, PA 19117

Using slides the presenters will share several uses of the computer for enhancing lessons in an integrated curriculum unit for middle school students focusing on the theme of medieval life. Partici-pants will learn ways to utilize a computer for CAI and as a tool including word processing and data base activities.

Specific Objectives:

1. To explain a school district project which involved the use of computers to enhance the planning and implementing of an integrated curriculum unit related to one theme....medieval life.

2. To present software programs for use in a medieval life unit at the middle school level.
3. To explain related writing, word processing, and data base activities related to a medieval unit.

Description of the Format:

A brief overview of the district's integrated curriculum unit related to the theme of medieval life will be presented. It will show:

1. How a social studies teacher worked with colleagues in planning and implementing integrated lessons on medieval life that involved students in much reading, writing and computer usage;

2. How the reading and English teacher planned reading, literature and writing units related to the theme of medieval life;

3. How the teacher of the gifted used a word processor to plan medieval book talks, gave these talks and engaged in storytelling using computer graphics;

4. How teachers used a data base program to plan their efforts to incorporate the arts into the content areas focusing on the theme of medieval life.

The presenters will then:

a. Present ways to use selected software programs to the theme of medieval life;

b. Present software that can be used for a specific unit such as that of medieval times; and

c. Explain writing, word processing, and data base activities related to a medieval unit.

ABSTRACT: Development and Integration of Computers into the Curriculum

Mary Ann Coe, Archdiocese of
Philadelphia, Philadelphia,
PA 19103

This presentation will center on the fevelopment and implementation of Computer Education Guides written for grade one through eight. The discussion will focus on understanding the computer as a tool so as to integrate its use into the existing curriculum of all subject areas. The speaker will explain how these guidelines are being developed, piloted, and implemented.

ADMINISTRATIVE USES

John A Bartley
W. Edward Bureau
Julianna Csongor
Richard Luoma
Linda Dean
Paul McEnerney

ABSTRACT: Strategic Planning for the Use of Microcomputers in Instructional Programs

John A. Bartley, W. Edward Bureau,
Springfield School District,
Delaware County, Philadelphia,
PA 19064

Rather than merely placing computers in the schools with the hope or suggestions that teachers use them, Springfield School District has systematically made computers an integral part of curriculum and instruction. Although strategic planning has brought about this success, three unique approaches have been used; each has had distinctive yet successful components.

The Deliberate Decision Approach: A decision to develop, fund and implement a Computers Applications course was made solely at the District level. Once an action plan was drawn up and District funding assured, teachers were solicited, curriculum written, and inservice training provided before and during the year of implementation. Results are obvious in a popular and successful general education course taught by teachers from several subject areas. From inception to implementation of the course, a one year time period was needed.

The Evolutionary Approach: Over a multi-year period, computers have been introduced into a science department in a phased, evolutionary process designed to gradually increase levels of teacher comfort, knowledge, and skill with computers. Accompanying the process was a continuing increase in the amount and sophistication of hardware and software. Concurrent staff education efforts included presentations on weekends and professional days; awarding teachers minigrants; and conducting inservice courses. Successful progress in this approach is evident not only by the level infusion of computers in curriculum and instruction but also by the creation of "teacher experts".

The Intensive Project Approach: A three year plan to introduce computers into language arts instruction at the high school became intensive in nature with funding from a Chapter II grant. During the second year of the plan, a grant project was designed and implemented to directly meet needs in curriculum, instruction, and student writing performance. Within a one year period, a word processing lab was set up; teachers received intensive instruction in teaching writing with word processors; and curriculum was radically revised. Evaluated at the end of its year, the project evidenced pronounced effects in all three areas as well as in teacher feelings of ownership. Refining of teaching skills, student writing, and curriculum continue during the current year.

Presenter's Note: The intent of the presentation is to briefly describe the three approaches within the context of the District and its managerial procedures; to report the status of success of each; and to offer additional resources and information to those listening to the presentation.

ABSTRACT: Student Records at Your Fingertips with Database and Spreadsheet Software

Julianna Csongor, Little Flower
Catholic HS, 10th & Lycoming
Ave, Philadelphia, PA 19140

Teachers having to face a multitude of individuals daily, trying to make decisions in their best interest often need fast and accurate background information on students.

Database and Spreadsheet software provide the means of placing relevant student records literally at our fingertips on the departmental level. Each of us input and update our own students' records but we find the investment in time required for this a small price to pay for having information readily available when needed.

We accomplished our basic goal using PFS File on a Radio Shack TRS-80 Model 4 computer and are investigating how we could improve on it by switching to Profile 4 Plus on the Model 4, or dBASE III, or Lotus 1-2-3 on a Tandy 1000. A report on the present status of our experiments will provide a comparison of these softwares.

ABSTRACT: Administrative Uses of the Computer at the Building Level

Richard Luoma, Linda Dean,
Paul McEnerney, Hatboro-Horsham
S.D., 229 Meetinghouse Road,
Horsham, PA 19044

The presentation deals specifically with PFS Write, PFS File, and PFS Report with one application of Graphing Assistant. The first application uses biographical data from a data base to generate lists using PFS Report for almost any need which may arise during the school year. The second part of the presentation shows how to integrate data from PFS File with a letter written on the word processor and produce mailing labels.

The third segment demonstrates an accounting file designed on the data base to keep a record of money pledged and collected by hundreds of students for a major school project. The presentation will conclude with a specific use of Graphic Assistant to show a grade distribution of math grades in our middle school.

INDUSTRY AND PUBLIC LIBRARIES

John DiLullo
Marguerite Buck

ABSTRACT: The Philadelphia School District/Local Industry Cooperative Training Project

John DiLullo, Supervisor, Division of Computer Science Technology, School Dist of Phila., 21st & Parkway, Room 507, Philadelphia, PA 19103

Since its inception four years ago, nearly one thousand School District of Philadelphia teachers have been exposed to the computer and introduced to its capabilities through the support of local corporations and the people who make up these corporations.

Dudley Cooke, General Manager of Information Systems for the Sun Company, recognized the need for the infusion of technology into the Philadelphia School System and approached Dr. Constance Clayton, Superintendent for the School District, with the proposal for lending Sun Company staff, resources and executive support for the purpose of helping teachers adapt to computers and technology. Volunteers from the Sun Company, Cigna Corporation, Mellon Bank, Smith Kline Beckman then attended sessions familiarizing them with the types of computers and software used in the Philadelphia School District. Training was coordinated through the Division of Computer Science Technology and the Office of Staff Training for the School District of Philadelphia.

From the very beginning, this program has been unique in that it provides human resources rather than the hardware and software implementations that usually characterize industry/school cooperative programs. Rather than providing the hardware, which is of little value without trained people to use it, this program provides the more valuable, although intangible, resources that make for successful classroom implementations with computers - time and people.

The program has expanded to include a half-dozen other corporate groups in the 1986-87 school year and has become a powerful vehicle for the delivery of general computer knowledge and use of software applications to the classroom teachers in Philadelphia.

ABSTRACT: Establishing a Microcomputer Center

Marguerite M. Buck, Chester County Library and District Center, 400 Exton Square Parkway, Exton, PA 19341

In October 1984, Chester County Library received a Library Services and Construction Act grant of $52,820. The purpose of the grant was to promote computer literacy for adults within the County and to train librarians and library staff on the microcomputer.

A librarian was hired to implement the grant. Microcomputers, printers, software, and furniture were selected (see attached list). In January of 1985, the Microcomputer center was opened in the Library in what was two closed classroom areas. Within the Center, there is software to introduce the patron to the IBM PC computer, tutorials on popular software programs, word processors, databases, spreadsheets, graphics programs and games (see attached list). A one hour orientation session is offered on a regular basis as an introduction to the equipment and the Center.

More in-depth training is offered through hands-on computer literacy workshops. To date, fifteen, 9 hour workshops were offered at the Library. In addition to the computer literacy workshops, word processing workshops using PFS Write were also well received.

In November 1985, the PC Users Group held its first meeting at the Library. This group has grown from 14 in attendance to a main meeting with 110 participating in November 1986. Some topics covered during a main meeting were artificial intelligence, desktop publishing, public domain software, and specific software programs. A question and answer session is always included and a card file of experts is available. Special Interest Group meetings include Prolog language, C language, Lotus 1-2-3 and Dbase III. Although this club was started by the Computer Services Librarian, it is working toward being an independent organization with library support. The Club acquired and keeps updated the PC Blue library of public domain and shareware software.

Future plans include a computer Literacy workshop for local policemen, and a study to assess the feasibility of circulating software, as well as continuation of existing services.

This project won the following awards:

National Association of Counties 1986 Achievement Award Better Business Bureau Award 1986 for Most Dramatic Growth of a Service, nonprofit category

EARLY CHILDHOOD

Eric S. Heller, George Forman
JoAnn Springsteen, Dene G. Klinzing
Susan W. Haugland, Daniel D. Shade

ABSTRACT: Young Children, Robots, and Logo: Cooperative Problem Solving in Three-Dimensional Microworlds

Eric S. Heller, The Institute for Governmental Services, University of Massachusetts, and George Forman, School of Education, University of Massachusetts, Amherst, MA 01003

In this study of young children's cooperative problem solving behavior, computer-driven robot toys are used to create observational environments that are both highly motivating and engaging for children, and easily manipulated and controllable for the investigator.

Robotic environments are created through the integration of Robotix toys, a microcomputer, and the Logo programming language. The toys include building pieces and bidirectional 3-volt DC motors, combined to create a multifunctional robot, consisting of a wheel-driven chassis and a three-dimensional gripping arm. Through a customized interface, the motors are activated and deactivated by memory-address value adjustments. Logo is used for software interface development, as its precedure-oriented nature lends itself to the creation of routines uniquely linked to the functions of the robot.

To stimulate cooperative play in pairs of young children (ages 5-7), observational environments are created in which unique roles are delineated for each child and for the computer. Three variables of robot operation are recognized: the limb to operate next (sequence), the direction in which that limb will move (initiation and direction), and the decision to stop that limb's motion (termination). Sequence is built into the program as a recursive loop, controlling for that element of cooperative problem solving. Initiation/direction is assigned to one child, while the second controls termination. The children are asked to maneuver the robot, pick up an object and deposit it at a third location. The course of their collaboration is recorded on videotape for later analysis. The analytical model through which the tapes are viewed involves the construction of ideal solutions to the objective that are compared with the strategy employed by the children. At points of discrepancy, analysis is made of gestures, conversation and decisions in an attempt to decipher cognitive processes involved, both at the individual level and in the cooperative mode.

ABSTRACT: Teaching Young Children to Use Computers: Techniques That Work

JoAnn Springsteen, M.S., and Dene G. Klinzing, Ph.D., University of Delaware, Alison Hall, Department of Individual & Family Studies, Newark, Delaware 19716

This presentation describes young children's successful use of computers in a model preschool equipped with computers and in specially designed computer camps. The successful integration of computers into the lives of these children can be attributed to the application of developmental principles throughout the curriculum.

The connection between developmental theories and computer curriculum will be examined. Specifically, important techniques that can be useful to all those who work with young children and computers will be paralleled with the theory on which it is based, to provide a framework for designing current and future curriculum development. These techniques include: (1) constructing an interactive learning environment; (2) providing an atmosphere that is conducive to independent exploration; (3) providing opportunities for additions, manipulations and reworking of already acquired knowledge; (4) encouraging repeated use of skills and actions; and (5) anticipating the sequence of the learning stages. Examples of how these techniques are used in actual computer curriculum will be provided.

ABSTRACT: The Effect of Developmental Software on Young Children

Dr. Susan W. Haugland, Associate Professor, and Dr. Daniel D. Shade, Assistant Professor, Department of Home Economics, Southeast Missouri State University, Cape Girardeau, Missouri 63701

The KIDS Project was organized to investigate the impact of microcomputers on early childhood education. Many teachers and parents have accepted the computer as an educational tool for young children without questioning whether the software currently being marketed fits children's developmental needs. Furthermore, research has not answered critical questions regarding computer instruction in early childhood education. For example: What kinds of computer experiences are beneficial? Does computer instruction enhance children's overall development? Can the computer teach children some skills more effectively than other early childhood curriculum?

The KIDS Project began to investigate these questions by establishing a scale for developmental software. Ten criteria were established to identify software that meets young children's needs. Commercial software was then rated using this developmental scale.

Software obtaining a score of five or above was made available to 40 children ages 4-5. Quantity of time spent using a particular piece of software was measured. These children were compared after eight months of computer exposure to 40 control children who had no exposure to computers. Differences in children's conceptual development, fine motor skills, perceptual discrimination, social interaction and positive self-concept were measured by the McCarthy Scale of Children's Abilities, the California Preschool Social Competency Scale and the Inferred Self-Concept Judgement Scale.

This presentation will provide an overview of the developmental scale and then briefly review the project's findings. It is expected there will be significant differences in children's development due to computer exposure as measured by the assessments.

K-8 WRITING

Judy Piper
Evelyn Bender
Gary A. Carnow

ABSTRACT: Using Computers to Improve Student Writing

Judy Piper, Alice Birney Elementary School, Sacramento, California

The Alice Birney Elementary School project is funded by the California State Department of Education under the Educational Technology Local Assistance Program. Funds provided for the expansion of the Alice Birney computer lab to 11 computers and four printers. This developmental grant emphasizes the use of computers as learning tools to improve student writing across the curriculum. The computers/word processors facilitate critical thinking by students during the writing process. In order to improve students' attitude toward writing, student work is published.

The fifth- and sixth-grade students receive weekly lessons using the computer during the writing process. Teachers stress written language activities and correlate them with all other curriculum areas whenever possible. Alice Birney students develop fluency by writing journal entries weekly on a variety of topics and by participating in response groups. They write monthly assignments on a seasonal theme, developing critical thinking skills during prewriting, writing, revising and postwriting activities. The stories/poems are published in class books in which each student has a page. Books are entered in a school-wide book fair, providing the authors with an audience of other students, teachers and parents.

The computers are also used to provide computer/writing training for all teachers and parent volunteers at Alice Birney School. A unique feature of the program is the use of parent volunteers to publish student work at all grade levels.

ABSTRACT: Reading/Writing/Research: Technology in the Service of Creativity

Evelyn Bender, Stetson Junior High School IMC, Philadelphia, Pennsylvania 19134

Reading/Writing/Research, a cooperative effort between a reading teacher and a librarian, has taught students to use word processing in their writing. The project was funded by PATHS (Philadelphia Alliance for Teaching Humanities in the Schools), a local consortium of foundations.

The teaching strategy was to demonstrate programs that progressed from graphics to word processing: *Print Shop*, a graphics program; *Story Tree*, which guides students into writing interactive stories; and *Magic Slate*, a word processing program.

The students worked very hard at writing. In addition to their scheduled time, they spent much of their free time in the library, working on writing and using the library's resources for research reports. The atmosphere of joyous learning, the pleasurable creativity, and the great extent of peer teaching that occurred exemplified an ideal educational situation. In addition, the project afforded the students a rare opportunity to do individual work. The students have been very enthusiastic.

The writing project has continued in its second year with the same students now in eighth grade. A new seventh-grade class has started on the cycle, repeating last year's activities. The project has been extended to several special education classes with funding for hardware and software provided by a grant from the Pennsylvania Department of Education.

Stetson Junior High School, the project site, is an inner city school with 1,600 students. The success of the program suggests that it can be replicated easily in other schools.

ABSTRACT: N.E.W. Visions--Noteworthy Experiences With the New Technology Language Arts Project

Gary A. Carnow, Ed.D., Coordinator of Instructional Technology Education, K-12, Alhambra School District, Alhambra, California 91801

The N.E.W. Visions Project is one of 13 Developmental Grants sponsored by the California State Department of Education. The project was funded to develop materials and lessons that integrate computers and video into the existing language arts curriculum.

The project involves classrooms in two elementary schools. In 1985-86, teachers in grant classrooms wrote a series of technology-enriched language arts activities and gained proficiency in *FrEdWriter*, a public domain word processing program that features "prompted writing."

In 1986-87, students were involved in a four-month study. Students experienced six weeks of daily keyboarding instruction in a lab setting and a four-month writing program integrating computers and video technologies. Control groups experienced the same language arts program without the technology-enriched environment.

The results of this study will help answer the following questions:

- Does technology motivate student behavior in the language arts?
- Does student writing improve when computers replace traditional paper and pencil tasks?
- What keyboarding skills are gained by fourth-grade students who receive 30 minutes per day of keyboard training for six weeks?

The data from this California state project provides timely information to districts wanting to:

- Implement technology at various levels;
- Embark on meaningful technology staff development; and
- Develop an "enriched curriculum" within current curricular offerings.

TEACHING ALGEBRA WITH THE SUPPORT OF TECHNOLOGY

Richard Carter, *Chair*
Lesley College
Cambridge, MA

ABSTRACT

This project developed materials to teach introductory algebra to sixth grade students. Two algebraic microworlds were developed and tested: a Marble Bag Icon Laboratory and an Algebra Workbench. The Marble Bag Icon Laboratory was used to aid in the development of the concepts of unknown, variable, and equation. The Algebra Workbench, a beginning algebra expert system, was used to aid students in the formal algebra manipulation tasks. In addition, Logo programming was used to allow students to construct their own mathematical machines for a variety of tasks such as generating algebra quizzes and coding and decoding messages using algebraically-based coding schemes.

The materials were taught to two classes of sixth graders during the 1986-87 school year. Results thus far suggest that sixth graders are able to learn algebra when mathematical concepts are "hooked" to their world of experience, made visual through interactive icons, and aided by the Algebra Workbench.

The panel consists of curriculum developers, researchers, and classroom teachers.

PANELISTS

Wallace Feurzeig
Bolt Beranek and Newman
Cambridge, MA

Judith Montgomery
Berkeley High School
Berkeley, CA

John Richards
Bolt Beranek and Newman
Cambridge, MA

Nola Sheffer
Hanscom Middle School
Bedford, MA

SPONSOR: SCS

GRADING PROGRAMMING CLASSES

Harriet G. Taylor, *Chair*
Louisiana State University
Baton Rouge, LA

ABSTRACT

One of the major problems faced by the computer science teacher is evaluating students, particularly in courses which concentrate on programming. Teachers must decide on how to give partial credit, how to count documentation, how to control cheating, and how to define and determine an acceptable program. Tests must be written to measure the skills the students are expected to acquire. Instructors are then held accountable for the grades that are assigned.

The panel members will discuss the various issues involved in grading programming classes. They will discuss guidelines and formats that are followed at their respective institutions. The session will conclude with a group discussion with the audience.

PANELISTS

Joyce R. Brennan
The University of Texas at Austin
Austin, TX

J. Philip East
University of Northern Iowa
Cedar Falls, IA

David Olson
Western Oregon State College
Monmouth, OR

Marguerite K. Summers
University of North Carolina - Charlotte
Charlotte, NC

SPONSOR: SIGCSE

TEACHER TRAINING

Elizabeth Uzdavinis Carlson
Janet Graeber, Mary Furlong
Suzanne C. Joseph
Ida Malian

ABSTRACT: The Impact of the Training Modules for Trainers Project on Training

Elizabeth Uzdavinis Carlson, Associate Director, Project TMT, The University of Michigan, Ann Arbor, Michigan

School districts' investment in computers has created a need for inservice training of school personnel in appropriate computer use. The Training Modules for Trainers (TMT) Project, funded by the Michigan Department of Education, has developed materials designed to meet the unique needs of teacher inservice trainers.

A "trainer of trainers" model has been implemented in the development of the modules and in the dissemination of training. Using this model, project staff have conducted workshops for 200 trainers in the state. This group has then used the modules to train key people in districts throughout the state of Michigan, who have, in turn, trained teachers and administrators. The impact of this model is being evaluated and will be used to highlight significant implications for the design of future training.

The *Training Modules for Trainers* series consists of 14 modules, each of which addresses a major aspect of educational computing. Each contains an overview, goals, prerequisites, competencies, issues, references, activities, blackline masters and feedback forms. The modules have been designed for use by both experienced and inexperienced trainers as a resource to provide additional support for training. New trainers can rely on the modules to provide a framework for workshops while experienced trainers use the materials as a reference.

The modules serve several purposes. They not only provide an organized and relevant resource for trainers, but they also suggest specific strategies for planning and conducting training. The ultimate purpose is the effective integration of computers into schools.

ABSTRACT: Preparing Teachers for the Information Age: The University of San Francisco Teacher Credential Program

Janet Graeber, Program Associate, University of San Francisco, Ignatian Heights, and Dr. Mary Furlong, Director of Teacher Education, University of San Francisco, Ignatian Heights, San Francisco, CA 94117

At the University of San Francisco, future educators are learning to effectively use the technology of our rapidly changing world. These new tools are being integrated throughout the teacher education curriculum. In accordance with the California guidelines for teacher preparation, the following computer competencies are included in the USF program:

1. Operation of computer based instructional systems;
2. Evaluation of software;
3. Use of computer applications (e.g., word processing, electronic networks/telecommunications, graphing, spreadsheets, and data base management);
4. Creative use of the computer as a tool for problem solving and critical thinking; and
5. Awareness of the societal impact of computers.

Through classsroom observations, student projects, and hands-on laboratory experience, these future educators use computer application programs as part of their regular coursework. Questioning strategies that emphasize higher level cognitive skills are demonstrated through instruction and practiced by students at their off-campus field sites. They learn pedagogy that incorporates the latest technology, focusing on the teacher as the learning facilitator.

To facilitate observations of exemplary teaching, USF has established a partnership with the Schools of the Sacred Heart in San Francisco. Their faculty model the roles of teacher as instructional decision maker, learning facilitator, classroom manager, and life-long learner, while exposing future educators to technology integrated into the K-12 curriculum. As part of the partnership, these faculty members instruct in the Teaching/Learning Lab and the Clinical Observation/Tutoring classes, thereby bringing the latest trends in teaching practice to the next generation of educators.

ABSTRACT: Implications of Cognitive Style for Computer Implementation in Middle Schools

Suzanne C. Joseph, Doctoral Candidate, College of Education Studies, Drexel University

In keeping with the recognized role of the teacher in the education process, this presentation will focus on my findings as to the relationship between the cognitive style of the teacher and the implementation of computers. In doing so, I will examine the implications this relationship has for a new instructional delivery system: an information system made up of humans and computers.

As with the implementation of any information system, the extent of user participation actually realized in organizational efforts remains unsatisfactory. Zmud suggests the existence of a link between cognitive style and attitude toward information system use as a way in which the user's ability to participate can be studied.

The concepts and methods derived from work on cognitive styles have also been applied to educational research. Such applications have included studying how students learn, how teachers teach, how teachers and students interact as well as interests/choices within educational-vocational domains (Witkin, Moore, Ottaman, Raskin & Karp, 1977).

The central research hypothesis of my study is that a relationship exists between the cognitive style (as operationally defined by the Embedded Figures Test) of the middle school teacher and the implementation of computers for instructional use: such that the pattern of implementation of computers for instructional use varies with respect to the teacher's cognitive style. The independent variables of this study are: cognitive style, either field dependent or independent; and sex. The dependent variable is divided into subcategories of instructional uses of computers (e.g., drill and practice, word processing, programming, etc.).

ABSTRACT: Computer Literacy as a Function of Computer Aptitude and Interest Among Preservice Teachers

Ida Malian, School of Teacher Education, San Diego State University, San Diego, California

The purpose of this research was to determine the relationship between tested computer aptitude and tested computer literacy. The underlying contention was that preservice education students may possess computer aptitude but not actualize this aptitude. Consequently, the same students score low on given computer literacy tests. This contention was supported by the data in this research. The data further demonstrate a significant relationship among experienced literacy and aptitude. Differences between male and female students and differences among specific education majors (elementary, secondary, special education and physical education) were also found. Teacher training options and exit competencies for computer literacy will be discussed.

BITNET AND BEYOND: NATIONAL AND INTERNATIONAL ACADEMIC COMMUNICATIONS ON YOUR CAMPUS

Michael D'Amore
EDUCOM
Princeton, NJ

ABSTRACT

Access to a growing worldwide network of educators, scholars, researchers, and administrators is now within the grasp of virtually every US institution of higher education. BITNET has made such connectivity possible at low fixed costs, allowing campus-wide participation across all disciplines, at its over 200 US member institutions. This session will describe BITNET and the services provided by the BITNET Network Information Center as well as other networking options for campuses wishing to join this growing electronically interconnected community. Future plans for BITNET and other academic-oriented networks in the US will also be described.

SPONSOR: EDUCOM

SOFTWARE FOR THE HUMANITIES CLASSROOM

Nancy M. Ide, *Chair*
Vassar College
Poughkeepsie, NY

ABSTRACT

A variety of programs and systems are under development for use in teaching humanities disciplines. This session is designed to give an introduction to three such programs, each of which is intended for use in a different field. Each of the programs discussed demonstrates an imaginative use of computer technology to enhance instruction in a particular discipline, and each goes far beyond computer use for "rote" learning and drill and exercise. The intention is not only to familiarize attendees with the particular programs that are discussed, but also to acquaint them with new possibilities for innovative use of the computer in humanities education.

(1) *The Computer in the Philosophy Classroom*
Almost all philosophy departments (and many mathematics and computer science departments) regularly offer a symbolic logic course or two. Two of the most difficult topics for students in a typical introductory symbolic logic course are (i) the construction of proofs, both in sentential logic and in predicate logic, and (ii) the symbolization or translation of natural language sentences into a symbolic language for first-order predicate logic. While some students have trouble with some other topics in a symbolic logic course, many more have trouble with these two, or at least some aspect of these two. This talk describes and demonstrates two computer based tutors being developed at the Center for Design of Educational Computing at Carnegie Mellon intended to help students master these two difficult topics. While both tutors are currently being developed for advanced function workstations, plans exist to port them to less powerful personal computers, such as the Macintosh and the IBM PC.

(2) *The MUSE Piano Practicer*
The MUSE Practicer provides an intelligent environment for students to practice piano playing. The system provides examples of correct performances, evaluation of the student's performances, and playback capability. The practicer by itself is a supplement to teaching piano; in the future, we intend to augment the system with an intelligent piano tutor to provide instruction tailored to each student's ability.

(3) *Simulations in US History*
Simulations in US History is a series of programs designed to place the student in historical settings and put him or her in the role of the decision-maker. The student is asked to answer a specific question—for example, should the southern states be allowed back into the Union after the Civil War?—and in order to answer, the student has to use the system to gather information, explore various options, and decide which among various paths of questioning to follow.

PANELISTS

Leslie Burkholder (1)
CDEC/CMU
Pittsburgh, PA

Roger Dannenberg (2)
Carnegie-Mellon University
Pittsburgh, PA

Jordan M. Smith (3)
University of North Carolina
Chapel Hill, NC

SPONSOR: ACH

COMPUTER-BASED TUTOR ON GRAPHING EQUATIONS

David Trowbridge
Jill Larkin
Carol Scheftic

Center for Design of Educational Computing
Carnegie-Mellon University
Pittsburgh, PA 15213
(412) 268-8527

Abstract

We describe the development of a computer-based tutor concerned with the graphing of simple algebraic equations. The program has some of the attributes of an "intelligent" tutor, including a model for problem solving, a coach for helping students to apply the problem solving model, a facility for entering arbitrary new problems within a class, and the capability of solving any problem in that class. Known as Sketch, it teaches a systematic approach to curve-sketching. This paper details our steps in designing, implementing, testing and improving the program.

Introduction

Computer Assisted Instruction today presents new options and opportunities: powerful machines, advanced computer science techniques, and input from cognitive psychology. Using these tools well is a challenge for which there are few guidelines or examples. The purpose of this paper is to describe how we used some simple artificial intelligence techniques in a powerful educational programming environment to capture some of the expertise of experienced teachers in an instructional program that runs on microcomputers.

Overview of the Program

Sketch helps students to develop skills of visualizing and quickly sketching graphs of simple algebraic equations. It guides them through the transformation of simple expressions into more complex ones, and the transformation of curves, one step at a time. It emphasizes transformations of the shape and location of curves, rather than the plotting of points. At present, Sketch handles simple expressions, such as $y(x) = 2\sin(3x)$), that contain a single non-arithmetic function. It can guide the graphing of any expression of this type. Sketch is a program of considerably greater flexibility than is usual in computer based instruction. It features:

- A model for problem solving that is suitable for a class of problems -- the utility of the model is not limited to any particular collection of problems.
- A coach for helping students to apply the problem solving model -- the coach is able to give appropriate help at each step.
- A collection of instructive examples which is readily expandable by the teacher -- problem sets can be designed to reinforce the current lesson or to review previous lessons.
- A facility for entering arbitrary new problems -- students can enter their own problems, provided those problems meet certain criteria of form and complexity.
- An interactive guide to using the program -- no separate instructions or documentation are required for effective use of the program.

The same problem solving model and coaching strategies are used with any problem posed. All problem expressions are cast into an internal representation that is suitable for any problem that meets the necessary criteria, so that the program is able to diagnose errors and give detailed suggestions, regardless of whether the problem was among the set of original examples provided by the program, added to the program by the teacher, or entered by students while using the program.

Description of the Program

The program is divided into three major parts. An interactive introduction to the program uses a simple example to introduce the problem solving strategy to the student. Upon entry, the student who has never used the program before is walked through a problem, following the problem solving model in a highly constrained manner which does not allow any deviations from the correct series of steps. Extensive guidance is given, concentrating on how to use the program, rather than on the concepts that will be taught later.

After finishing the walk-through problem, students are on their own. They may work problems from an existing set or enter their own problems. After selecting a problem, students are free to choose any steps they like. A coach watches each step of the student's solution, allowing several valid alternative paths at each step and giving context-specific help whenever the student starts down an invalid path. Figure 1 shows the appearance of the screen during a typical interaction.

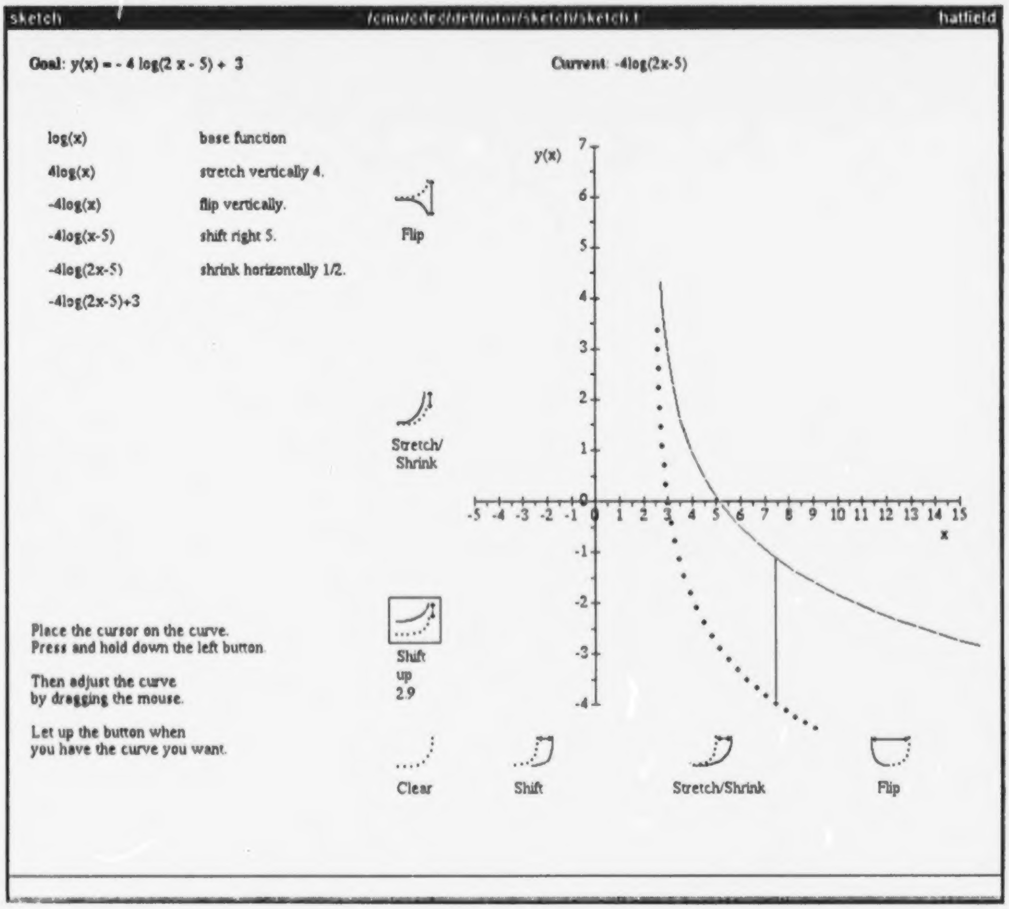
Problem Solving Model

The problem solving model is the explicit method used by the program to solve a problem itself. The program teaches this model by advising or correcting students when their work fails to follow the model. The major intelligent feature of the program is its ability to independently solve any problem of the specified type, and to use the resulting solution as a basis for coaching.

The first step in the problem solving model is to examine the problem expression and identify the *base function*. For example, the base function of the expression, $2x^3$-5 is x^3. (Although input of this function is currently in the form, x^3, future system developments will let the student use more natural notation.) The program currently recognizes the following base functions: x, x^2, x^3, sqrt(x), $\log_{10}$(x), ln(x), exp(x), abs(x), sin(x), cos(x), tan(x), arcsin(x), arccos(x), and arctan(x). Its architecture could readily support additional functions if desired. Once the student identifies the base function, the computer displays a graph of it.

Suppose the student goal is to sketch the expression: $y(x)=-4\log(2x-5)+3$. Following the problem solving model, the student identifies the variable and the base function, log(x), and the computer plots it.

The next step is to identify a single arithmetic operation that, when applied to the function or to the argument of the function, will result in a slightly more complex expression that advances toward the desired (complex) problem expression.



Goal: $y(x) = -4\log(2x-5) + 3$

Current: $-4\log(2x-5)$

log(x)	base function
4log(x)	stretch vertically 4.
-4log(x)	flip vertically.
-4log(x-5)	shift right 5.
-4log(2x-5)	shrink horizontally 1/2.
-4log(2x-5)+3	

Place the cursor on the curve.
Press and hold down the left button.

Then adjust the curve
by dragging the mouse.

Let up the button when
you have the curve you want.

Figure 1. Screen display midway through a student's solution to a problem.

The student modifies either the base function or the variable, by adding, subtracting, multiplying or dividing by numbers, or by changing the sign, one step at a time. At each step, the student must perform the corresponding geometric transformation of the curve, shifting it up, down, left or right, or stretching it vertically or horizontally, or flipping it about one axis or the other.

When the student has chosen an appropriate arithmetic operation, say replacing the base function log(x) with 4log(x), the program asks the student to perform a corresponding geometric operation on the displayed graph. For example, replacing the base function log(x) with 4log(x) corresponds to stretching the curve in the y-direction by a factor of 4.

One possible sequence of steps appears in Figure 1. Other orders of steps would be equally acceptable, as long as each algebraic step corresponds to a single, simple geometric transformation.

The model is flexible in that at many points in a solution there are at least two steps that can appropriately be taken. Because the program is based on a general problem-solving model, and not on any recorded sequence of steps, these alternative orders are automatically acceptable to the program. The problem solving model merely requires that the student make the algebraic expression more complex, one step at a time, and then, for each step, apply the corresponding geometric transformation to the curve.

The model can be described algorithmically by:

```
(1) Identify a base function.

(2) Plot the curve for the current expression.
    (This is done automatically.)

(3) Repeat until the current expression is equivalent
    to the given problem expression:

    (a) modify the expression by exactly one step:

        {[(Add, Subtract, Multiply, Divide)
        a number (to, from, by)], Change the sign of}
        the {current function, independent variable}

    (b) transform the curve in a way that corresponds
        to the algebraic transformation:

        {[Shift (up, down, left, right),
        Stretch (vertically, horizontally)] by a number},
        {Flip (vertically, horizontally)}
```

When we give demonstrations of this program, we are often asked how we developed the strategy it incorporates. The next section describes that process.

Design Process

The first development step was an extensive effort to import the knowledge of teachers with experience in mathematics and science and with an interest in innovative approaches to teaching and learning. We began by asking the superintendents of local public school districts to nominate their top mathematics and science teachers at the secondary level for participation in this project. We then hosted an open house for the nominees, at which we displayed the computing environment we would be using, and several examples of educational applications.

We identified six teachers who appeared particularly qualified. Our selection criteria included a combination of classroom teaching and individualized tutoring experiences, curriculum planning experience, and evidence of the use of innovative approaches.

Two mathematics teachers participated in subsequent design sessions for the program described here; the other four teachers helped us design two other programs.

Preliminary Design Session

The two teachers shared a variety of experience teaching geometry, advanced algebra, trigonometry, and calculus. We spent the first morning brainstorming areas where students might benefit from an additional learning tool, without regard to the technical details about how such a tool might be implemented. Topics suggested during this session ranged from graphing skills to the meaning of the derivative, but there appeared to be a common theme that emphasized the relationship between the algebraic and the graphical representations of a particular problem or expression.

By mid-day, we had reached agreement that we would focus on a tool concerned with the rapid sketching of algebraic expressions. We spent the afternoon in determining a vocabulary of relevant issues, a description of the problem solving strategy to be implemented, an overview of necessary modules, and a diagram of the relationships among the components. We selected two sample problems and considered specifically how we would handle them in this context, concentrating on the pedagogical issues of the interaction between the tutor and the student.

The First Prototype

Before the next group meeting, the two teachers prepared twenty questions each, of the type that they would want this system to be able to handle. Using a highly interactive programming environment (see Programming Environment below), we quickly generated some graphical displays as candidates for eventual use in our program. Using the teachers' sample questions, we investigated how they might be presented in our displays.

Subsequent Design Sessions

Our next set of meetings covered two consecutive days. We began by reviewing our understanding of the system to be designed, based on our previous meeting. We promptly moved into detailed discussions of the interactions that might take place as a student proceeded through several of the sample problems.

We started with a sample problem and spent most of the first day examining how our system might handle it. The teachers discussed how they would present the sketching of this equation if they were individually tutoring a student. We tried to relate their suggestions to our sample displays, modifying the visual representations as necessary to meet the conditions the teachers considered most important. We discussed the system's responses to both correct and incorrect input from the students, noting and diagramming each step.

Next, we went through the same steps for several additional problems. We tried to use the same process, the same form of input and responses, for several different types of problem. When we found a situation that was not covered adequately by our original plan, we modified the plan so that it would cover both the new problem and all previous ones.

Program Development

The design sheets from our three days of work with these expert teachers can be viewed as what cognitive psychologists call a protocol -- a detailed, on-line record of the thinking of these expert teachers as they designed a lesson (Newell & Simon, 1972; Anderson *et. al.*, 1984).

From such design sheets it is possible (and often appropriate) simply to write an instructional program that implements sequentially the interactions described in them (Bork, 1985). Viewing the design sheets as a record of expert teacher thinking, however, we took a somewhat different approach. We designed a program to capture the knowledge of the teachers in a way that could generate a lesson on any of a broad range of problems. The lessons on the design sheets were then naturally generated by this general program, but the program would be able to handle a large class of additional problems as well.

This approach also simplifies the task of providing appropriate feedback to the student. Since our system can generate its own multiple versions of possible solution sequences, the program is able to give very specific responses to a wide range of different steps that a student might propose. This is possible, furthermore, regardless of whether the problem under consideration is one of the system's standard examples or whether it was proposed by the student.

Field Testing

Initial testing was done by making the program available to a variety of users of CMU's Andrew system and then modifying the program based on informal feedback. A more systematic test was conducted in the Fall of 1986, when the program was made available to students at a local high school for two days. Detailed observations were made, followed by appropriate modifications.

As a result of both the formal and informal testing, we have revised many messages, redesigned the user-interface, and speeded up some display routines. It is interesting to note that, due to the extremely short revision/review cycle of our programming environment (see Programming Environment below), it was possible to make some small but crucial changes during the five minute break between class periods at the high school, and to use the revised program with students immediately!

Implementation

Implementation of these ideas required design of a suitable data structure. As is often the case, a good choice of data structure is central to the program's performance. In Sketch, expressions are represented internally as a tree. The nodes of the tree correspond to the base function, the independent variable, and the operators that act on them.

How the Data Structure is Used

The diagram in Figure 2 illustrates the tree as it would be set up for the expression $-4\log(2x-5)+3$. In the tree, the next appropriate steps are always the adjacent nodes either upward or downward from the current function. As can be seen in Figure 2, after sketching $y=\log(x)$, the student could modify it to produce either $y=\log(x-5)$ (one down in the tree) or $y=4\log(x)$ (one up in the tree). If the student chose to sketch $4\log(x)$, then the following step could be either $-4\log(x)$ or $4\log(2x-5)$, moving one up or one down from the subtree for the preceding expression. The system also recognizes the special case of unary minus (changing the sign, flipping the curve), where an operation two nodes away from the current function is considered a valid next step.

Representing the equations as trees has the following advantages. First, any equation with a single base function can be represented with equal ease simply by making a smaller or larger tree. Second, this data structure lets the program easily find the next step(s) in the problem-solving model. Therefore it can either offer suggestions or, if the student is really stuck, execute the step. Third, the tree naturally supports a variety of possible correct solutions because it always lets the student take any appropriate next step. Finally, by finding exact discrepancies between the tree and students' proposed algebraic steps, very specific feedback can be given.

The following sample interactions illustrate Sketch's ability to respond to errors. The student is working toward the final equation $y(x)=-4\log(2x-5)+3$, and is entering the step following the base equation $\log(x)$. The program is using the tree in Figure 2. Each interaction is followed by a summary of the program's process for recognizing the student's error. In all cases the program inserts numbers and variables appropriate to the current example.

Student entry: $-4\log(2x-5)+3$

Program response: That's the whole equation, but we're not ready to make such a complex change from the previous equation.

Process: *It compares the student's expression with the complete problem expression.*

Student entry: $\log(2x-5)$

Program response: You have indicated two operations on the variable: subtracting 5, and multiplying by 2. Please choose one or the other.

$y(x) = -4\log(2x-5) + 3$

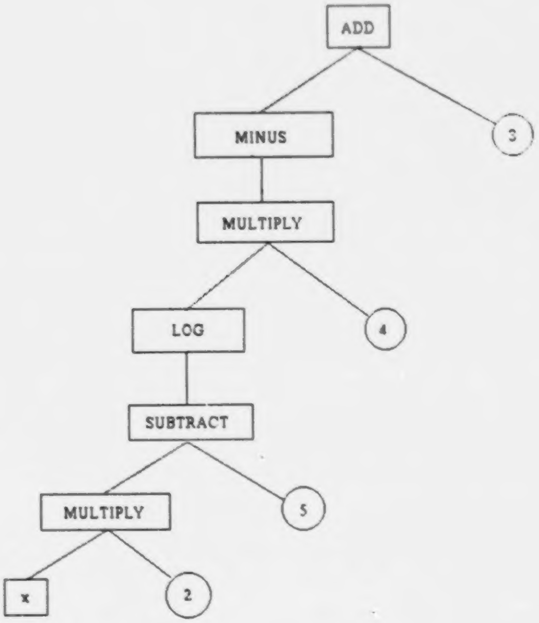


Figure 2. Problem representation using a tree data structure.

Process: *It "walks the tree" down from the base equation node (log) two levels (subtract and multiply). The student expression is identical to the sub-tree of the problem expression that is spanned downward from the base function to the independent variable.*

Student entry: $\log(x-5)$

Process: *It recognizes this as a legitimate next step, and hands control over to a routine for getting the corresponding geometric transformation*

Implications of Using a Tree

A tree data structure like that used here is common in computer science, but less common in CAI. Because this data structure does not have a predefined size, but may be larger or smaller depending on the problem, getting information from the tree requires a recursive technique to do depth-first search. This technique, easily implemented in most languages (Winston & Horn, 1981), looks through a tree, to whatever depth is necessary, to find any specified item that the program currently needs.

Although the tree is a fine data structure for the program, this is not how people write equations. Therefore, to let both students and teachers define their own problems, we have built a parser that takes an equation in conventional form and builds the tree that the program needs.

User Interface

For **Sketch** to be an easy-to-use aid in thinking about curves, it requires a display that reflects the problem solving model and that makes the program's operation self-evident. The algebra interface is reasonably straightforward, with the student simply typing in successive expressions. The geometry interface, on the other hand, requires a special device that makes it easy to move or reshape curves.

A set of six transformation icons are displayed; three correspond to shifting, stretching and flipping horizontally; three more correspond to performing those transformations vertically (about the x-axis). Upon selecting one of these icons, the cursor changes to cross-hairs which may be positioned anywhere on the curve. By pressing the left mouse button and dragging the mouse, the student transforms the curve. In shift mode the curve keeps its shape as it moves up, down, left or right. In stretch/shrink mode, the curve expands or contracts relative to one of the axes. In flip mode, it switches between y(x) and -y(x) or between y(x) and y(-x).

After several iterations of the user-interface, we feel that the current scheme gives the student the closest feeling yet of direct manipulation of the curves.

A two-column table records the history of algebraic steps and corresponding geometric transformations. The desired generality of the program required us to develop algorithms for producing nice looking axes with reasonable labels, numbers and tick marks. Designing the screen was aided by ports, rectangular sub-regions within the window. Ports are subroutines containing information about how the display should behave if the user changes the shape of the window. During design, ports can be rearranged on the screen to improve the display without losing their properties.

Workstations

Certain characteristics of advanced computing technology were central to the development of this program. Advanced workstations (e.g., the IBM RT PC, the DEC VAXstationII, and Sun Microsystems' Sun-2 and Sun-3) provide the following essential features: (1) virtual memory, which allows the programmer to work largely as if the machine had almost unlimited memory; (2) a high-level operating system (in our case, BSD 4.2 UNIX $_{\text{tm}}$), which provides both development tools and the potential for easily moving programs from machine to machine without modification; (3) a large bit-mapped display (like that of the Macintosh, but having about four times as many dots), which makes it much easier to display a large amount of information in a clear, logical fashion. The value of this last feature is illustrated in the figures of our **Sketch** program, where the student has simultaneous views of a developing list of equations, an interactive graphing environment, and instructions and coaching from the program. Features like this require about 3 megabytes of memory and a processing speed of roughly a million instructions per second.

Programming Environment

This program was implemented using CMU Tutor, a programming environment for advanced-function workstations which includes a number of features especially valuable in the development of educational software (Sherwood and Sherwood, 1986). CMU Tutor is a descendent of the TUTOR and MicroTutor languages originally developed in the PLATO project at the University of Illinois. With this history, many of its features have had 20 years of testing for effectiveness and efficiency.

CMU Tutor gives authors full access to the Andrew system, an educational computing environment being built at Carnegie Mellon University. Andrew is the distributed personal computing environment being developed in a joint project between IBM and CMU (Morris, *et. al.*, 1986).

The language is particularly well-suited for non-expert programmers. In addition to all of the usual control structures of a general-purpose programming language, CMU Tutor supports many capabilities which are particularly germane to the creation of educational applications. These include picture drawing, graph drawing, animation, answer judging, menus, mouse input, and display of fancy text.

CMU Tutor is an incrementally compiled language, making possible an extremely short revision/review cycle. Even with large programs, an author need never wait more than about 5 seconds to see the effects of changes just made.

The complex interactive interface developed for **Sketch** could not have been done in any reasonable manner without this set of tools. Implementation of **Sketch** has involved programming by three different people. Presently, the program consists of 7000 lines of code.

Acknowledgments

This program is part of a project funded by the National Science Foundation (Grant Number MDR-8470166) to explore the use of advanced function workstations for educational computing, with emphasis on teaching topics in science and mathematics to high school and college students. The project is based at Carnegie Mellon University; it represents a collaboration between CMU and the Allegheny Intermediate Unit (a regional educational service agency for schools in Allegheny County, PA). Additional support has been provided by a fellowship from the John Simon Guggenheim memorial foundation and the NSF program of Visiting Professorships for Women (Grant Number R11-8600412).

The authors wish to acknowledge the contributions of several additional people in the development of this program. David Yee has done an excellent job programming, and has had many good ideas for improving the user-interface. Bruce Sherwood and Judith Sherwood, our colleagues at CMU, patiently offered much assistance as we worked our way through the more technical details of CMU Tutor. Our two subject matter experts, Connie Radocay, a teacher in Plum Borough School District, and Andy Tiglio, a teacher in Chartiers Valley School District, provided both insight and inspiration.

Bibliography

Anderson, J.A., Boyle, F. C., Farrell, R. and Reiser, B., "Cognitive Principles in the Design of Computer Tutors," *Technical Report*, Department of Psychology, Carnegie Mellon University, Pittsburgh, 1984.

Bork, A., *Personal Computers for Education*, Harper and Row, 1985.

Erikson, K.A., & Simon, H. A., *Protocol Analysis: Verbal Reports as Data*, MIT Press, 1984.

Morris, J.H., Satyanarayanan, M., Conner, M.H., Howard, J.H., Rosenthal, D.S.H.. and Smith, F.D., "Andrew: A Distributed Personal Computing Environment," *Communications of the ACM*, 29:3 March, 1986.

Newell, A. & Simon,H. A., *Human Problem Solving*, Prentice Hall, 1972.

Sherwood, B. A., and Sherwood, J. N., "CMU Tutor: An integrated programming environment for advanced-function workstations," *Proceedings of the IBM Academic Information Systems University AEP Conference*, IBM Academic Information Systems, San Diego, April, 1986.

Sherwood, B. A., and Sherwood, J. N., *The CMU Tutor Language, Preliminary Edition*, Stipes Publishing Company, 10 Chester Street, Champaign, Illinois 61820, 1986.

Sherwood, B. A., "Workstations at Carnegie Mellon," *Proceedings of the Fall Joint ACM-IEEE Computer Conference*, November, 1986.

Trowbridge, D., "Using Andrew for Development of Educational Applications," *Proceedings of the 1985 Academic Information Systems University AEP Conference*, IBM Academic Information Systems, Milford, CT, p. 85, June 23-26, 1985.

Trowbridge, D., "A Sampler of Educational Software at CMU," *Proceedings of NECC '86*, National Educational Computing Conference, International Council on Computers in Education, University of Oregon, Eugene, OR, p. 135 June 4-6, 1986

Winston, P. H. & Horn, B. K. P., *LISP*, Addison Wesley, 1981.

CAI UNITS ON GRAPHING WITH NON-GRAPHICS TERMINALS

Robert J. Lopez*
Department of Mathematics
Rose-Hulman Institute of Technology
Terre Haute, IN 47803

*formerly at Memorial U. of Nfld.
St. John's, Newfoundland, Canada

Abstract

As part of a remedial program in mathematics at the Memorial University of Newfoundland, three first year math courses were cast into a CAI format by the writing of 252 programs of total content ten megabytes. A number of units were drill and practice programs for graphing: conics, rational functions, polynomials up to fourth order, square root functions and their inverses, piecewise linear functions. The mechanics of creating such graphing lessons and delivering them on non-graphics terminals is discussed. The effect of such lessons is also described.

1. Introduction

Lopez, (in press) reports that the three-course remedial mathematics sequence for the first year at Memorial University of Newfoundland was successfully converted to a CAI format. This four year project addressed documented deficiencies (Gaskill and Lopez, 1978) in student performance at Memorial. To cover the sequence of a remedial, a precalculus and a first calculus course, a total of 252 lesson programs consisting of ten megabytes of code were written in the DECAL authoring language (Digital Equipment Corporation Authoring Language) for execution on eight dedicated hard-copy terminals (Teletype-43's). In spite of the drawbacks of noise, slow print speed (300 baud), and paper cost, the hard-copy format was necessary because of the length of the dialogues generated by these programs.

In each of the three courses there were units on graphing. Consequently, a number of drill and practice programs were needed to exercise students in the skills of graphing conic sections, polynomial functions up through fourth order, rational functions, square root functions and their inverses, and piecewise linear functions. The constraint of the non-graphics terminals was handled by a strategy of synthesis and analysis that proved to be more effective than any alternative that could have been devised using graphics-capable terminals.

2. Graphics Strategy

A careful study of the objectives of the units on graphing techniques led to a comprehensive classification of the possible graphs that the student would either need to recognize or sketch. Thus, for example, the conics were not rotated, and, except for the parabola, were centered at the origin; the degenerate case of a pair of lines was not considered. Consequently, for both the ellipse and hyperbola there were only two possible configurations for each of these two conic sections. These graphs are shown in fig. 1 where each such conic section is depicted with its unique code number. After deciding on what the graph of a particular conic section looks like, the student communicates this choice to the lesson program by entering this code number. This is how the computer program "knows" which image the student has selected as the corect representation of the graph of a given function.

A typical dialogue involving the ellipse would begin with the presentation of an expression such as

$$4x^2 + 5y^2 = 10. \qquad (1)$$

By a multiple choice format the student had to identify (1) as the equation of an ellipse. To ascertain that the student would actually be able to sketch its graph, the dialogue continued with the following questions, the very questions any tutor would ask in trying to draw out from the student the correct graph of (1).

a) What is (are) the x-intercept(s)?
b) What is (are) the y-intercept(s)?
c) From fig. 1, what is the coded designation of the graph that most nearly resembles the graph of (1)?

Because of the limitations of the DECAL authoring language the questions could not be posed exactly as indicated above. For example, there was no capacity for submitting and processing an answer in the form

(0,2) and (0,-2).

Hence, to avoid giving away to the student that there were two x-intercepts and two y-intercepts, and that at a y-intercept the x-coordinate was zero, the actual questions asked were in the form

a) How many x-intercepts are there?
b) What is the x-coordinate at an x-

intercept?
c) What is the y-coordinate at that x-intercept?

This more cumbersome dialogue would not be needed with a more flexible authoring system such as the new DEC Course Authoring System (CAS) and is therefore only an accidental consequence of the authoring language used.

It is not immediately apparent from the simplicity of the above example that the student needed both to analyse and synthesize information contained in equation (1). First, enough detail had to be extracted from the equation to determine which of the two possible configurations for the ellipse was right. Next, the student had to synthesize a conceptual representation of the correct ellipse so that a match between the appropriate graph in fig. 1 and the abstracted version of the given conic could be made. The efficacy of this combined analysis, synthesis, and matching is more obvious in the next example of a parabola.

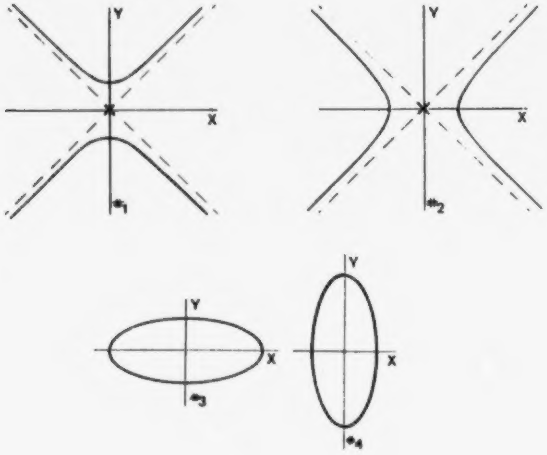


Figure 1
The centered ellipse and centered hyperbola. Note the simple coding used here to distinguish one graph from another.

3. Parabolas

The equation of the parabola could be presented in any one of the forms

$$y = a x^2 + b x + c \tag{2}$$

$$a x^2 + c = y + b x \tag{3}$$

$$x(-b - a x) = c - y \tag{4}$$

with the coefficients a, b, c being randomly generated real numbers such that a could not be zero.

Again, by multiple choice format, the student had to identify the given equation as that of a parabola, then complete a dialogue resulting in the analysis needed for sketching an appropriate graph. The student, therefore, had to find the roots of the quadratic equation $y = 0$, the x-intercepts (if any), the y-intercept, and the vertex, and then determine whether the graph opened upwards or downwards. From this information the graph of the parabola had to be synthesized and its representation found in a chart containing 26 possible configurations of such a parabola. Fig. 2 shows a portion of the chart available to the student for this purpose. Each parabola is identified by its unique code number. In the event that the student had to identify the chart on which the correct graph was to be found, the various charts were indexed also. Instead of the tedium that might first be expected, this need to select a correct chart forced the student to categorize and name the graph being searched for and to contrast it with alternative graphs with respect to intrinsic properties. The net result was a considerable increase in the skills of graphing.

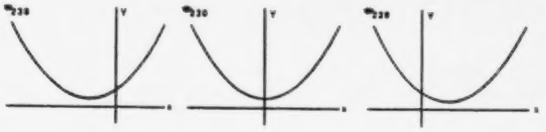


Figure 2
Three of the 26 possible parabolas. The code numbers incorporate five properties that distinguish one graph from another. The coding is implemented at run-time and is governed by base-three arithmetic.

For example, the function

$$y = x^2 + 2 x + 2 \tag{5}$$

has the complex zeros $x = -1 + i$ and $x = -1 - i$, and its graph has no x-intercepts, a y-intercept at (0,2), a vertex at the point (-1,1) and opens upwards. Hence, graph number 238 in fig. 2 is a representation of the graph of (5).

Note that the determimation of image number 238 as the correct representation of (5) requires that the student clearly articulate the distinguishing characteristics of this graph. Thus, the student must recognize that the vertex is in quadrant 2, that there is no x-intercept, that the curve lies above the x-axis, opens upwards, and has a positive y-intercept. All of these features must be deduced and articulated if the correct graph number 238 is to be obtained.

4. Effectiveness

No formal experiment was designed around the CAI courses because the project was initiated to determine the parameters for such an extended experiment. This is a common feature of initial CAI efforts, the MIME project (Bajpai, et al., 1984) being another example where this happened. Generally, the computing resources for a large educational experiment are not available. In order to acquire them, administrators require evidence that the project is reasonable. Hence each project is forced to acquire its own benchmark data in order to qualify for more formal proposals.

The project at Memorial was no exception. However, at Memorial, the students in the CAI classes were in parallel with classes taught in the traditional mode. All students took the same commonly devised and administered panel-graded final exam. That the CAI students exhibited greater mastery of the skills of mathematics is anticipated in the literature (Bork, 1981; Chambers and Sprecher, 1980; Dalton and Hannafin, 1984; Dence, 1980; Fahey, 1984; Kearsley, Hunter, and Seidel, 1983a, 1983b; Leiblum, 1981; Lower, 1980; Moonen, 1980; Poirot, 1980) where CAI has been found to be a viable mode of skill development. Specific details of the success of the Memorial University project have been presented by Lopez, (in press). In addition, cursory inspection of examination papers revealed that the CAI students consistently scored higher on the graphing questions than their counterparts in the traditional classes. It is indeed unfortunate that these results were not obtained under controlled conditions.

5. Mechanism

The mechanism whereby the data base of possible graphs was organized and assigned numeric codes is an interesting exercise in base-three arithmetic that is applicable in more complex situations. Thus, the method used in the programs described here can be extended to more complex structures.

For example, parabolas of the form (2) fall into 26 distinct classes characterized by the five properties: location of the y-intercept, vertical location of the vertex, horizontal location of the vertex, number of x-intercepts, and direction of opening. Each of these five properties is assigned a unique digit a_k, $k = 0, \ldots, 4$, as follows:

a_0 = 2,1,0 [y-intercept negative, positive, zero respectively];

a_1 = 2,1,0 [vertex below, above, on x-axis respectively];

a_2 = 2,1,0 [vertex left of, right of, on y-axis respectively];

a_3 = 2,1,0 [no, two, one x-intercept respectively];

a_4 = 2,1 [upward or downward opening respectively].

The graph code is then the base ten equivalent of the base three number $a_4a_3a_2a_1a_0$, namely,

$$a_4 3^4 + a_3 3^3 + a_2 3^2 + a_1 3 + a_0.$$

This code was run-time generated from the parameters a, b and c of (2) and did not depend on visualization of the parabola. Thus, a_4 was 1 or 2 depending on whether the coefficient a in (2) was negative or positive, while a_3 was 2, 1 or 0 depending on the value of the discriminant $b^2 - 4ac$. This scheme, with appropriate modifications, was sufficiently robust to support graphing of the conic sections, rational functions, piecewise linear functions, third and fourth order polynomials, and square root functions with their inverses.

Specific details, program listings, etc., can be made available to interested readers.

6. Conclusions

Providing drill and practice in graphing in a CAI format for a remedial sequence of mathematics courses posed a severe programming challenge, one magnified by a restriction to non-graphics terminals. The solution adopted proved to be effective. In fact, the efficacy of the graphing procedure adopted appears greater than that of alternative procedures implementable on graphics-capable terminals.

The method of having students analyse the function to be graphed by extracting relevant details is typical in any approach to graphing. Having the students then synthesize these details into a conceptual whole is the goal of all instruction in graphing. Requiring the student to select an appropriate representation of the curve from an exhaustive listing of possibilities forces the student to make feature-by-feature comparisons which results in a greater awareness of the various properties of the graphs and functions in question.

The device whereby the list of alternative graphs is codified proves to be flexible and robust enough to support a wide range of graphing exercises. The graphing exercises created for the project described herein met every requirement of the existing syllabi for the courses they served. Had these syllabi been any more demanding, the programs could have been written to satisfy these requirements also.

The success of the CAI students vis-a-vis the traditional students taking the same final examinations indicated the efficacy of the analysis/synthesis provoked by the lesson programs. This result is consistent with the common observation that CAI is most effective in inculcating specific and identifiable skills.

References

Bajpai, A. C., Fairley, J. A., Harrison, M. C., Mustoe, L. R., Walker, D., and Whitfield, A. H. (1984). The MIME Project at Loughborough - a First Report. Int. J. Math. Educ. Sci. Technol., 15, 781.

Bork, A. (1981). Learning with Computers. Bedford, Massachusetts: Digital Press.

Chambers, J. A., and Sprecher, J. W. (1980). Computer Assisted Instruction: Current Trends and Critical Issues. Communications of the ACM, 23 (6), 332.

Dalton, D. W., and Hannafin, M. J. (1984). The Role of Computer-Assisted Instruction in Affecting Learner Self- Esteem: A Case Study. Educational Technology, 24 (12), 42.

Dence, M. (1980). Toward Defining the Role of CAI: A Review. Educational Technology, 20 (11), 50.

Fahey, P. J. (1984). Learning About Computerized Instruction with Adults: One School's Trials, Errors, and Successes. Educational Technology, 24 (7), 11.

Gaskill, H. S., and Lopez, R. J. (1978). Let's Bring Back Subtraction. Int. J. Math. Educ. Sci. Technol., 9, 221.

Kearsley, G., Hunter, B., and Seidel, R. (1983a). Two Decades of Computer Based Instruction Projects: What Have We Learned? T.H.E. Journal: Technical Horizons in Education, 10 (3), 90.

Kearsley, G., Hunter, B., and Seidel, R. (1983b). Two Decades of Computer Based Instruction Projects: What Have We Learned? T.H.E. Journal: Technical Horizons in Education, 10 (4), 88.

Leiblum, M. (1981). Structuring Computer-Assisted Learning in a University Environment. In Rushby, N. (Ed.), Selected Readings in Computer-Based Learning. London: Kogan Page, Ltd.

Lopez, R. J. (in press). Computer Delivered Remediation Based on the Hand-Held Calculator. Int. J. Math. Educ. Sci. Technol.

Lower, S. K. (1980). Computer Assisted Learning in Context. In Rushby, N. (Ed.), Selected Readings in Computer-Based Learning. London: Kogan Page, Ltd.

Moonen, J. (1980). The Teaching of Statistics and CAL. In Lewis, R. and Tagg, E. D. (Eds.), Computer Assisted Learning: Scope, Progress and Limits. Amsterdam: North-Holland.

Poirot, J. L. (1980). Computers and Education. Manchaca, Texas: Sterling Swift.

COMPUTER GRAPHICS FOR DIFFERENTIAL EQUATIONS

Sheldon P. Gordon
Suffolk Community College
Selden, NY 11784

Abstract:

The present paper will discuss the use of computer graphics programs in teaching an introductory course in differential equations. The author will consider the use of a series of programs he developed to provide graphical representations for many of the standard topics, including the convergence of the Euler approximations to the solution of a first order differential equation, the solutions of the second order linear, constant coefficient homogeneous differential equation, the tangent field for the solutions of any first order differential equations, Frobenius series approximating solutions and the use of differential equations as models in such areas as logistic or inhibited growth and predator-prey population analysis.

Introduction:

In recent years, the rapid growth in the availability and low cost of modern computers have brought these devices into the mathematics classroom. Computers are being used widely and successfully in the CAI mode in algebra and other remedial classes. Their use, particularly in the graphics mode, is increasing rapidly in calculus, precalculus, statistics and probability to enhance student understanding of topics and methods through visual impact.

Another mathematics offering which can be significantly enhanced by the introduction of computers to emphasize understanding of the concepts is the introductory course in Differential Equations. Unfortunately, very little effort has been made in this direction to date. Most introductory Differential Equations courses seemingly present the student with a collection of special techniques or "tricks" (such as integrating factors, undetermined coefficients, variation of parameters, etc) to solve a variety of specific types of differential equations on a purely manipulative basis. In addition, a modest amount of theory indicating existence and uniqueness conditions for solutions is interspersed. However, students in such courses rarely gain any deep understanding of the underlying ideas because these ideas are too thoroughly hidden by the manipulations and by the relative complexity of the solutions.

Several people have previously introduced the computer into Differential Equations courses, most notably John Van Iwaarden of Hope College and Howard Penn of the U. S. Naval Academy. However, the primary thrust of most of theese efforts has been in the direction of using computational programs to generate numerical or tabular solutions to the differential equations using a variety of numerical procedures. This approach is certainly desirable in the sense that it emulates the procedures used in practice to solve real-world differential equations that typically cannot be solved in closed form. However, while giving a practical flavor to the course, they do not do much to give the students a better understanding of the concepts. In order to do this, it is necessary to introduce geometric interpretations for most of the topics in the course. Unfortunately, most of today's students have very little geometric insight and have severe problems visualizing mathematical concepts. Therefore, the use of computer graphics provides an ideal way to encourage the geometric understanding of these concepts so that the students better appreciate what they are doing. At the same time, it is hoped that the use of the graphics will also assist them in developing better geometric insight.

Some Applications of Computer Graphics to Differential Equations

The present paper describes part of a comprehensive package of computer graphics programs being developed and used by the author which covers most of the important concepts and methods of an introductory course in Differential Equations. In particular, the programs developed utilize the graphical capabilities of the computer to demonstrate visually the geometric nature of the solutions, their dependence on initial conditions and their dependence on the appropriate parameters, coefficients and/or the functions involved. They are designed for instructor use in the classroom on a demonstration or exploratory basis as well as for individual student use in the Math Learning Center and in the College's Computer Learning Center in the library. Each of the programs is fully interactive in the sense that the user, either a faculty member or a student, is able to supply any desired function, any initial conditions, appropriate parameters or coefficients, and any interval on which the solution is to be displayed. The programs do not require any knowledge of programming beyond the ability to type in the desired functional expression according to BASIC syntax. The author has found that this can be covered in a matter of about ten minutes at the end of a class period, so that there is virtually no class time lost in the process of getting students involved in computer use.

While the primary motivation in developing these programs is to enhance student understanding of the subject, their availability also allows for some interesting flexability in the differential equations course itself. Based on the use of the software, it is now possible for the instructor to introduce any of three important new dimensions to the course. First, it is possible to introduce the students to a significantly deeper level of the theory of differential equations (admittedly only a surface introduction) which would normally not be considered until a follow-up course which addresses the geometric theory. Second, it is possible to emphasize some of the numerical ideas involved in approximating solutions of differential equations and therefore present some more realistic examples. Third, it is possible to emphasize the use of differential equations in the area of mathematical modeling where the computer plays the role of an experimental tool with which the students (or the instructor) can work to find the best fit between real-world data and the parameters (coefficients and/or initial conditions) in a differential equation. Some of the specific topics covered by this package are described below:

1. Euler Approximations: The fundamental existence theorem for the solution of the first order differential equation

$$y' = f(x,y)$$

is based on the idea of Euler approximations - a sequence of piece-wise linear approximations to the solution curve. At the same time, the method provides a reasonably useful introduction to the idea of approximating solutions of differential equations. Both of these points can be dramatically handled using computer graphics demonstrations to display the convergence of the Euler approximations to the solution of the differential equation for any reasonable function f, any desired initial conditions (x_0,y_0) and any desired interval over which the solution will extend. The program will draw, successively, the piece-wise Euler linear approximations based on n = 4, 9, 19, 39, 79 and 159 subdivisions of the indicated interval. As seen from Figure 1, it is visually evident that the successive approximations converge to a seemingly smooth curve representing the solution.

On the other hand, in cases where the Euler approximations do not converge because the corresponding conditions of the Theorem are not satisfied, the user is able to clearly see the divergent behavior of the successive "approximations" and therefore be better able to relate it to the chosen function.

2. Linear First-Order Non-homogeneous Differential Equations: Typically, in a differential equations course, the solution of the linear first-order non-homogeneous differential equation

$$y' = ay + g(x)$$

is found, if possible, using one of the tricks mentioned before. In the process of the manipulations, students invariably lose sight of what they are finding, namely a function that has particular properties that are tied to the differential equation as well as to the initial conditions. In this program, the user is able to construct a wide variety of functions for g(x) by combining choices of polynomial, exponential and trigonometric functions from a menu and then by specifying any desired coefficients and initial conditions. The program will determine the particular solution in closed form, draw its graph, and print out the expression for the solution. The program also allows the user to experiment with the effects of changes in the initial conditions on the resulting solution and its behavior.

3. Tangent Field: Given any first order differential equation y' = f(x,y), each point in the plane can be thought of as the initial point for some solution emanating from that point. Further, at each such point, the differential equation provides the slope y' of the tangent vector to the corresponding solution. The totality of all such tangent vectors is known as the tangent or direction

field for the differential equation. This concept can be demonstrated extremely effectively using graphics techniques to draw the tangent field of the differential equation $y' = f(x,y)$ for any desired function f and any desired two-dimensional "window" in the solution space. Specifically, for the indicated window (the range of x and range of y values), the program will automatically impose a pre-determined grid, use the differential equation to determine the slope of the tangent vector to the solution at each of the points of the grid, and draw the array of these vectors with the appropriate slopes. Moreover, the program also provides the capability to produce the graph of the approximate solutions from any initial point within the window.

In Figure 2, we show the tangent field for the differential equation

$$y' = x \sin(x+y),$$

an equation which cannot be solved in closed form. From a display such as this, it is simple to indicate how each initial point determines a different solution as well as to give a feel for what a differential equation really is. Further, it also becomes much easier to discuss the effects that the initial conditions have on the behavior of the solutions, thereby laying the groundwork for a discussion of more advanced ideas such as stability of the solutions. In this way, such graphs provide an outstanding opportunity to point out such concepts as domains of attraction and repulsion. These topics are usually delayed to far more advanced courses in Differential Equations because, without computer assistance, the students would not have the background to see the ideas. Yet, such notions become evident from displays such as that in Figure 2.

4. Logistic Differential Equation: One particularly nice application of differential equations is to the problem of inhibited growth as a more realistic model than the standard exponential growth situation studied in calculus. The graph of the logistic curve based on the logistic differential equation

$$y' = ay - by^2$$

is the typical S-shaped curve corresponding to the solution

$$y(t) = ay_0/[by_0 + (a - by_0)e^{-at}]$$

This topic can be effectively treated with computer graphics using a program that will draw the particular logistic curve corresponding to any choice of the coefficients a and b and any initial conditions. This program provides an ideal opportunity to explore concepts such as limits to growth and their independence from the initial population value as well as the notion of the equilibrium for such a population (and hence the equilibrium of the solution of any differential equation). Moreover, having this program available allows for student projects to explore the possibility of picking appropriate coefficients to match the logistic model to real-world data representing such phenomena as population growth, spread of disease, spread of a rumor, spread of a technological innovation, etc. Consequently, they gain an appreciation of what constitutes a good fit in mathematical modeling. Moreover, having this model available via a computer program allows the user to experiment with the effects of altering the terms slightly; for instance, by replacing the inhibiting term $-by^2$ with a different term, say $-by^{3/2}$

5. Second-order Linear Homogeneous Differential Equation: The graph of the solution of the second order, linear, constant coefficient, homogeneous differential equation

$$ay'' + by' + cy = 0$$

can be drawn for any choice of the three coefficients and any set of initial condition $y(t_0)$ and $y'(t_0)$. This gives the student the opportunity to see the effects of the initial conditions, particularly y_0', on the corresponding solution, as shown in Figures 3a and 3b. In general, in this course, we tend to describe the global behavior of the solution, but are unable to consider in detail the behavior of the solution in the neighborhood of the initial point. Such a program provides the ability to focus in closely on any such neighborhood to inspect the short-term behavior of the solution.

6. The Predator-Prey Model: The solution of the Lokta-Volterra predator-prey model of two interrelated species, say rabbits and wolves,

$$x' = ax - bxy$$
$$y' = cxy - dy$$

is drawn for any desired set of coefficients and initial populations. This program allows the students to experiment with the model by attempting to match actual data on such populations with the results predicted by the model based on different sets of coefficients. It also allows the student to vary the form of the terms in the model to see the effects of slightly different assumptions. Moreover, it is possible to graph the solutions in two ways - either x and y as two functions of the time t

(Figure 4a) or y as a function of x (Figure 4b). The latter approach allows for introducing the ideas of phase plane analysis and limit cycles.

7. Fourier Series: Graphs of the successive Fourier series approximations to any desired function on the interval $[-\pi, \pi]$ are drawn to illustrate the convergence of the sequence of Fourier approximations to the given function.

8. Frobenius Series Solutions: Power series methods are a valuable tool for obtaining approximate solutions to many differential equations for which closed form solutions cannot be found. However, the extensive manipulations involved are usually so complicated that most students completely lose sight of what they are deriving and how it relates to the differential equation in question. Computer graphics can be used to draw the graphs of the successive functions determined by the first n terms, n = 0,1, 2,3,4, of a Frobenius series solution to a differential equation. Such graphs clearly illustrate, first, that, each partial sum is indeed a function and, second, that this sequence of functions actually converges to the solution of the differential equation over successively larger intervals. In addition, such graphs demonstrate the geometric nature of a singularity for a differential equation. Such a display is shown in Figure 5.

In conclusion, the availability of these programs provides the students in our Differential Equations course with a much better and deeper conceptual knowledge of the subject. At the same time, they give the instructor the opportunity to open up a wide array of new topics, either in the direction of numerical approximations, more advanced theoretical notions or an introduction to the use of differential equations in a mathematical modeling sense. Needless to say, not all of this can be accomplished in any one semester. However, it does provide an effective way to give different flavors to the same course in different semesters. Finally, student reaction to computer graphics is always very favorable. Their level of interest can be held much more easily and their level of understanding is certainly improved.

Acknowledgements: The developments described in this paper were partially supported under grants from the State University of New York's program of Faculty Grants for the Improvement of Undergraduate Instruction and from the Tandy Radio Shack Educational Software Development program.

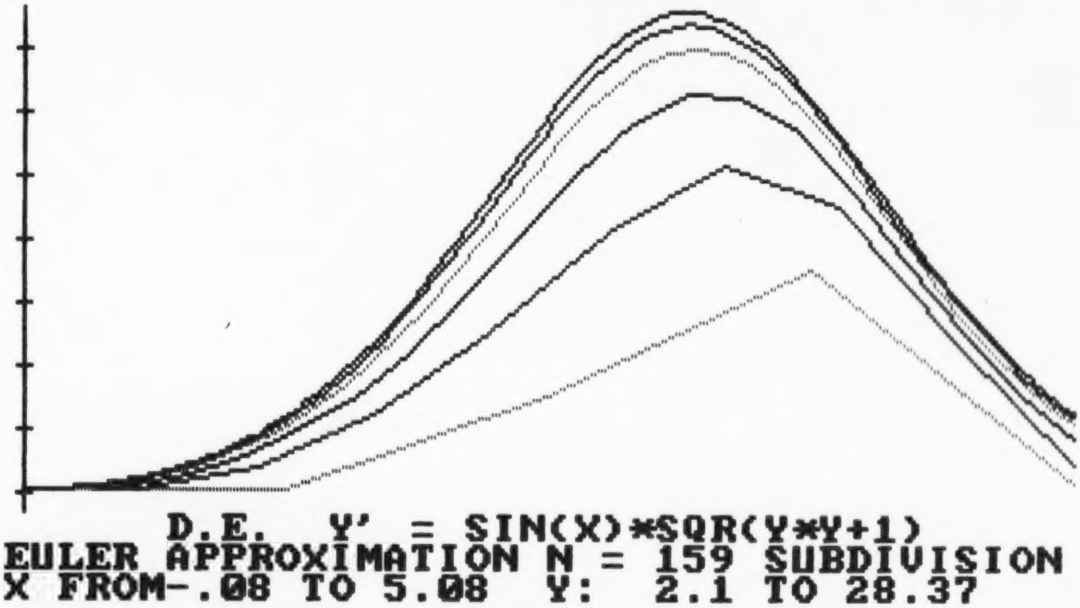
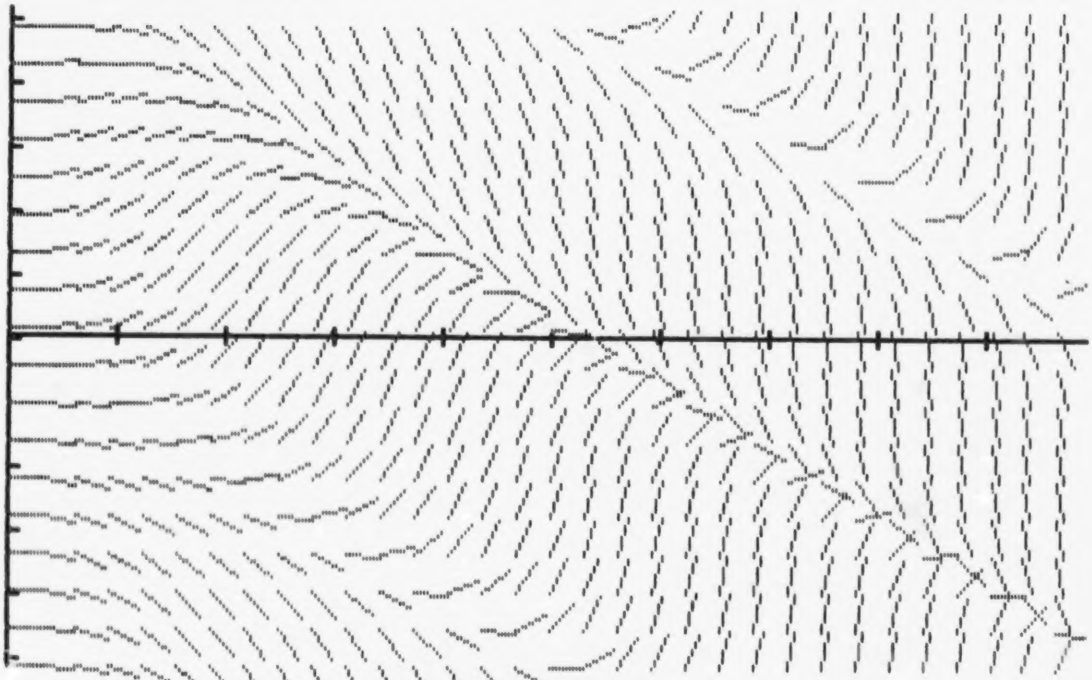


FIGURE 1



D.E. Y' = X SIN(X+Y)
X FROM-.01 TO 6 Y: -3.02 TO 3.02

FIGURE 2

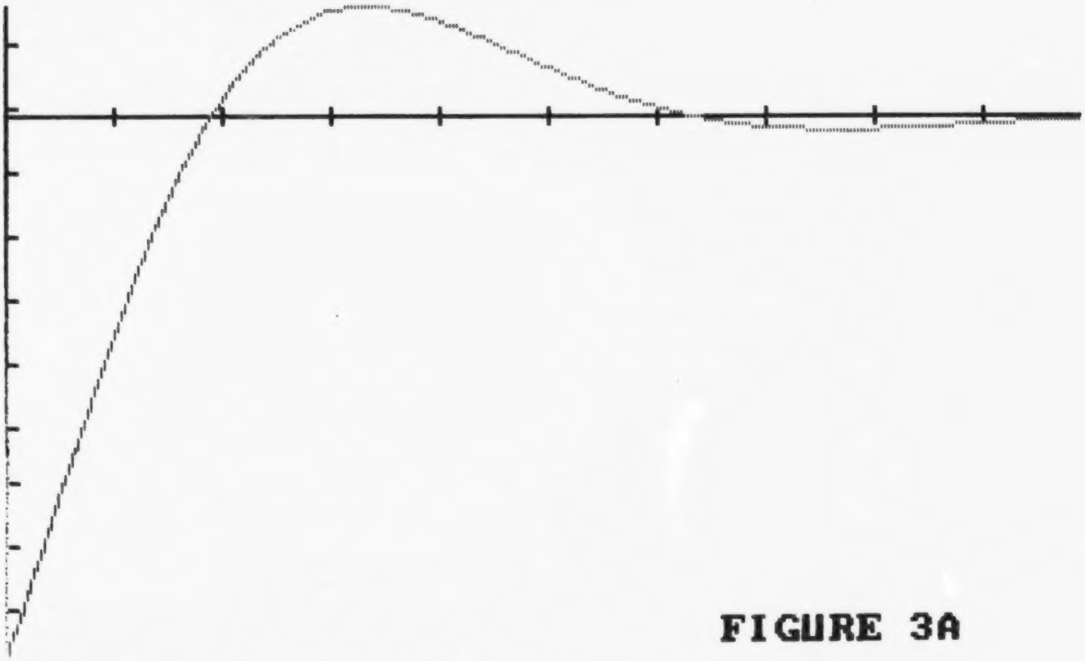


FIGURE 3A

SOLUTION OF 1 Y"+ 3 Y'+ 8 Y = 0

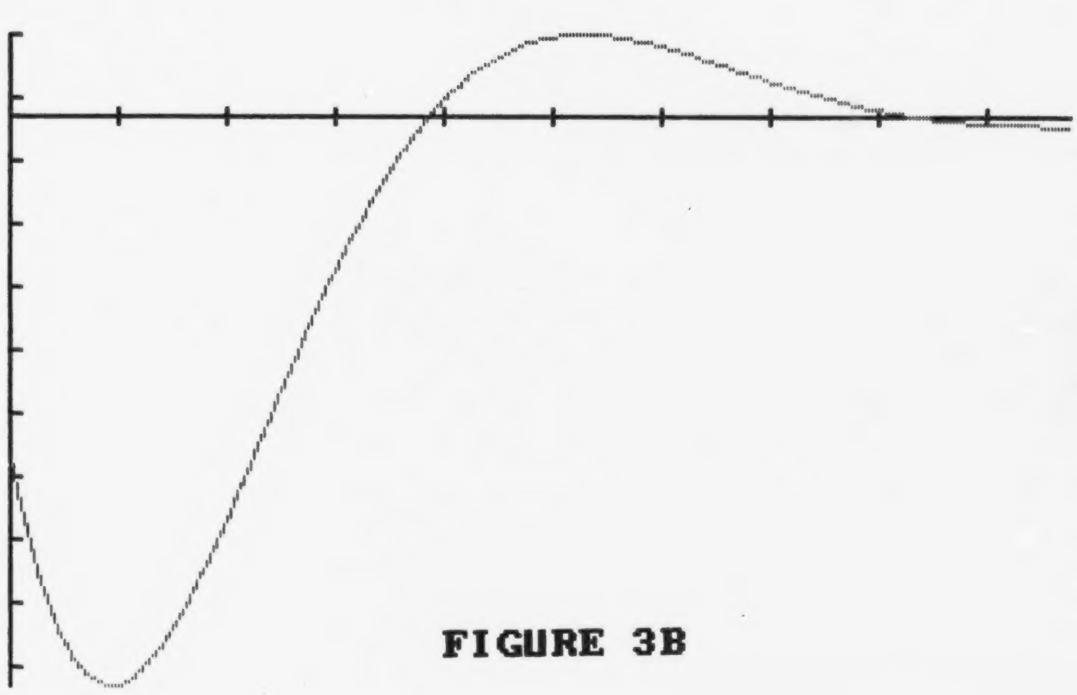


FIGURE 3B

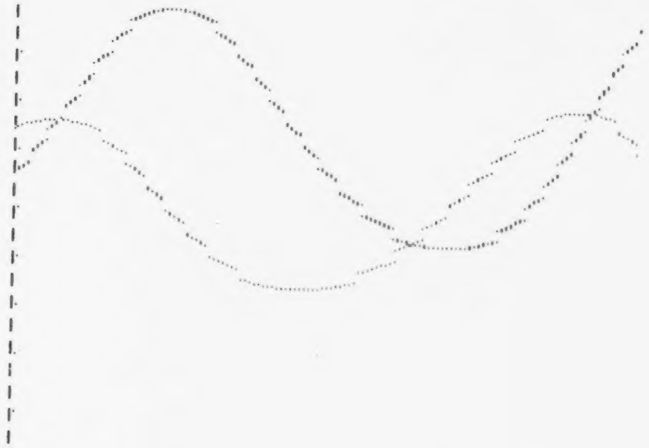


FIGURE 4A

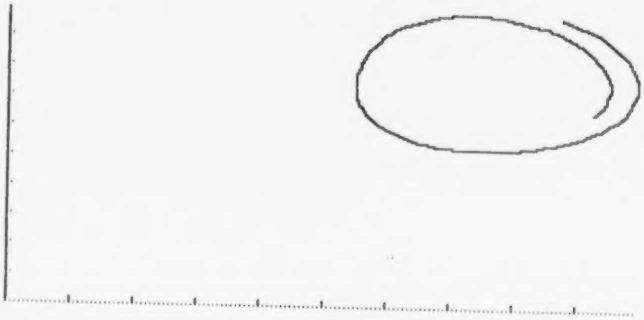


FIGURE 4B

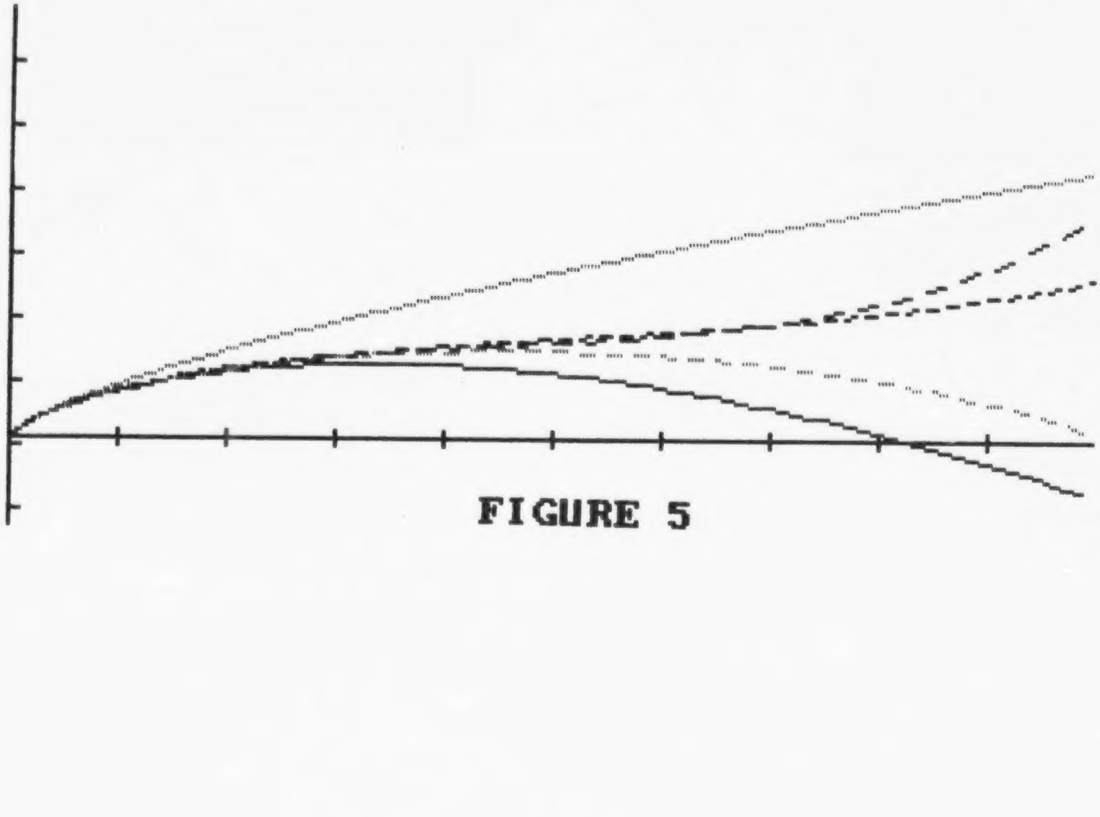


FIGURE 5

SOFTWARE LICENSING: A POSITIVE RESPONSE TO REAL WORLD PROBLEMS

Mark Friedland, *Moderator*
Krell Software Corporation
St. James, NY

ABSTRACT

Software licensing is a phenomenon that promises to make educational software accessible to millions of students. A handful of companies now provide plans whereby colleges and schools can afford to provide a program to every student. Many educators and computer firms are watching closely to see what benefits software licensing has to offer.

This panel will release the results of a mail survey exploring the current extent of software licensing use. Don't miss the latest news on licenses, and its significance to you!

PANELISTS

Steven Sumner
Interactive Education Foundation

Shirley Stas
Apple Computer, Inc.
Cupertino, CA

Karen Billings
Logo Computer Systems, Inc.
New York, NY

Pat Walkington
The Learning Company
Menlo Park, CA

Mont Francisco
Quicksoft, Inc.

Mike Burns
Ralston School District
Ralston, NE

Stuart Varden
Pace University

TEACHING AND USING APPLICATION SOFTWARE

Arthur Luehrmann, Chair
Computer Literacy
Berkeley, CA

ABSTRACT

Interest in computer applications is booming. In his 1985-86 survey of school computer use, Henry Becker found a sharp growth in teaching students about application software tools. In the 1986 examination of "computer competence" among U.S. students, conducted by the National Assessment of Educational Progress, questions were divided evenly between the topics of programming and applications.

Materials for teaching about applications are emerging. Recently published computer literacy textbooks offer whole chapters that describe word processing, data base and spreadsheet software. Still in short supply, however, are hands-on classroom materials aimed at building solid student skills in using these tools and applying them in the traditional curriculum.

The three panelists in today's session have been laboring for several years to create such materials. They will describe their reasons for doing so, their goals, and their views on the role of application software skills in both the computer curriculum and the traditional curriculum. In particular the panel will address these questions:

1. Why should anyone learn about word processor, data base, spreadsheet, graphics, communication, and other application software?
2. Who should learn about these topics? Who should teach these topics?
3. Is it enough that students have a general knowledge of computer applications, or should they also learn skills in using the software?
4. Does it make a difference which brand of application software the student learns?
5. Should application skills replace programming skills in the computer literacy curriculum?
6. How can computer application skills change what we teach and the way we teach traditional subjects?
7. What are successful strategies for integrating computer skills into traditional subjects?
8. If all students learn word processor skills, to pick one example, how many computers will schools have to own in order to handle the writing needs of their students?
9. What kinds of classroom materials do teachers need today?
10. What kinds of classroom materials will be needed in the future?

PANELISTS

Beverly Hunter
Targeted Learning Corporation
Amissville, VA

Stephen Marcus
South Coast Writing Project
University of California
Santa Barbara, CA

NSF ELEMENTARY MATHEMATICS MATERIALS GRANTS

Susan Friel, *Chair*
Lesley College
Cambridge, MA

ABSTRACT

The six recently funded NSF projects all focus on effectively integrating technology into the precollege mathematics curriculum. The grants are almost a year underway. Their project directors will present an overview of the problems each project is confronting and the strategies each group has developed for coping with problems in teaching mathematics. Facing the "crisis in education" is quite a timely topic. The six projects are: "Development of a Logo-Based Geometry Curriculum"; "Reckoning With Mathematics"; "Used Numbers"; "Calculators and Mathematics"; "K-6 Supplementary Mathematics Materials for a Technological Society"; "A Revision of the Geometry and Measurement Strands, K-6".

PANELISTS

Douglas H. Clements
Kent State University
Kent, OH

Glenn Kleiman
Education Development Center
Newton, MA

David Pagni
California State University at Fullerton
Fullerton, CA

Stephen S. Willoughby
New York University
New York, NY

William D. McKillip
University of Georgia
Athens, GA

SPONSOR: SCS

MATH/SCIENCE

Richard Carter
Daniel H. Watt, Molly Watt
Clarence DelForge, Milford Clark
David Sperber

ABSTRACT: Marble Bags, Balances, and Anti-Marbles: Approaching Algebra Through Concrete Metaphors and Graphic Images

Richard Carter, The Logo Algebra Project, Lesley College, Cambridge, Massachusetts 02238

We have found that many students experience algebra as a process of shoving around a set of abstract symbols according to a set of arbitrary rules. We believe that many of these students could benefit from a set of transitional ideas to help them move from the concreteness of arithmetic to the abstractness of algebra.

We have developed some experimental algebra curriculum materials that provide a series of concrete metaphors for algrebraic symbolization and manipulation. Part of this involves computer software that allows students to model algebraic operations through manipulating graphic objects and then supports the transition to standard algebraic notation.

In one of our teaching modules we use the following metaphors and graphic images:

1. A bag with an unknown number of marbles in it as a metaphor for an algebraic unknown;
2. A balance beam as a model of equality in an equation;
3. The idea of packaging equal sets of marble bags and marbles for factoring and distribution with parenthesized expressions; and
4. The idea of an "anti-marble" that cancels out a regular marble to model operations with signed numbers.

These ideas are introduced as tools to help students figure out secret number problems made up by themselves and other students.

We are developing software that will allow each of the models to be displayed and manipulated on a Macintosh using programs written in Logo. Through a series of interactive exercises and activities, students practice doing algebraic manipulations using the models. Once they become comfortable with these, they are introduced to standard algebraic notation as a shorthand way to record what they have already been doing.

ABSTRACT: Assessment of Critical Aspects of Logo Learning: A Collaborative Research Project with Teachers as Observers

Daniel H. Watt, Molly Watt, Education Development Center, Center for Learning Technology, 55 Chapel Street, Newton, Massachusetts 02160

Is the widespread use of Logo effectively promoting increased understanding of computer science concepts, problem solving strategies, or mathematical thinking among students? In many Logo classrooms, we observe that students have learned only the simplest aspects of Turtle Geometry and have not been exposed to some of the deeper and more powerful aspects of the language. Teachers often request support in evaluating their students' work and in finding ways to help their students get more out of Logo. Yet we often find that the teachers themselves are not fully aware of Logo's potential or how to tap it.

These are among the questions and concerns that have led to our project--a 15-month exploratory study funded by the National Science Foundation, starting in November 1986. We define "critical aspects of Logo learning" to be those aspects of Logo that are essential for its effective use but are not usually acquired by students without appropriate intervention by knowledgeable teachers. From January to June 1987, we will conduct a biweekly research seminar with a group of experienced Logo teachers in grades four through six. In collaboration with these teachers, we will identify and establish developmental sequences for critical aspects of Logo learning and develop a set of techniques for observation, assessment and enhancement of these aspects of Logo by classroom teachers.

Our project presentation at NECC will be our first opportunity to share preliminary findings with the computer education community. We will describe the critical aspects of Logo we have defined, share some of the insights about student learning that we have gained from teachers and outside observers, and report on the professional development of participating teachers.

ABSTRACT: Effects of CAI Math on Low-Achieving Students: An Eight-Year Longitudinal Study

Dr. Clarence DelForge, Dr. Milford Clark, Western Carolina University, Cullowhee, North Carolina 28723

A two-year study of CAI math (1976-78) was conducted in Jackson County, North Carolina with fourth-, fifth- and sixth-grade students at Log Cabin Elementary School, with the addition of Qulla Elementary School's third-grade students in the second year. The study was done in cooperation with the Elementary Education & Reading Department of Western Carolina University and Teacher Corps. They were trying to determine if CAI math would make a significant difference in student achievement in math for low achievers as measured by a standardized achievement test. The results were significant beyond the .01 level that the experimental group had greater growth than the control group.

As a follow-up to the original study, the students who had participated in the study were re-identified at the time of their high school graduation (1983, '84, '85, '86). Those who were still in the school system from the experimental and control groups were compared to see if early CAI math experiences made any differences on their high school scores on the North Carolina Competency Test of Mathematics. The major finding of this eight-year study was that CAI math instruction does make a difference in the achievement of low-scoring math students and probably has a lasting effect on their math skills.

It was also found that there were no significant differences between the experimental and control groups as to how many enrolled in high school computer classes and how many dropped out of school.

ABSTRACT: Computer Literacy as a Means to Academically Motivate Potential Dropouts

David Sperber, Palm Lakes Elementary, Miami, Florida,
Doctoral Student, Computer Education at Nova University

It is axiomatic that educationally disadvantaged students frequently find little satisfaction with routine educational and academic pursuits. It often happens that these students enter a downward spiral from which there is no escape. Poor grades result in low self-esteem and a disinclination to become involved in classroom activities, which, in turn, results in still poorer grades and further academic alienation. It is also understood that these same students often show increased interest and enthusiasm for hands-on learning tasks, especially when such tasks provide immediate feedback and reinforcement.

To capitalize on the interest of nonachieving students for concrete, hands-on learning activities, a computer literacy club was developed. The purpose of the club is to enable potential dropout students to master basic programming skills in elementary school and to help them understand how the development of such skills has direct application to a significantly large number of related computer careers.

Sixteen potential dropout students were selected. Primary critera for selection were: (1) at least one retention; (2) excessive absenteeism; and (3) poor academic performance. Students were informed that there were only eight "positions" available in the club. "Job interviews" were scheduled wherein students were required to demonstrate why they should be selected and what they could offer the Computer Literacy Club. Pretesting and posttesting were completed on the selected members. Preliminary results are most positive.

MOVING FROM THE BEACHHEAD TO INTEGRATION WITH PRODUCTIVITY TOOLS

David Weaver
Technology Program
Northwest Regional Educational Laboratory
101 S.W. Main Street, Suite 500
Portland, Oregon 97204
Telephone: (503) 275-9500

Abstract

This paper argues that the focus of computer literacy courses for the future should be upon the use of productivity tool software such as word processors, data base managers, and spreadsheets. The paper outlines the major advantages to such an approach and gives helpful tips to schools making the transition from a computer science based literacy course to one which focuses on productivity tools.

Introduction

Three years ago at the 1984 NECC preconference symposium on computer literacy, Arthur Luehrmann argued that the beachhead approach to computer literacy is more practical than an integrated approach. According to Luehrmann, just as an army begins an attack on an enemy territory by first establishing a beachhead from which to operate, so should schools when they first implement a computer literacy curriculum. In the beachhead approach, schools will appoint one teacher to be the computer teacher, place the computers in one location, develop a computer literacy course and acquire materials, and finally, run all of a particular group of students through the course. Courses developed under this approach tend to have a heavy emphasis on programming and computer science topics.

Beverly Hunter, who also spoke at the 1984 NECC preconference symposium, is a proponent of a more integrated approach. Ms. Hunter would rather have all teachers instruct their students on that aspect of the computer relative to their subject area. In this integrated approach, hardware is generally dispersed into the various classrooms or made transportable by placing computers on movable carts. The curriculum materials would vary according to each teacher and class, and tend to be open-ended, exploratory activities employing productivity tool software (word processors, data bases, etc.)

Much has happened since the 1984 symposium where such distinguished educators presented their arguments. As it happens, schools have generally elected for the practicality of the beachhead approach over the integrated approach. According to Henry J. Becker of Johns Hopkins University in "The Second National Survey of Instructional Uses of School Computers," programming and computer literacy remain the predominant uses of the computer, accounting for 48% of all use at the high school level, 28% at the middle/junior high school level, and 10% at the elementary level.

Although the beachhead approach is the one most commonly implemented by schools, it was never intended to be an end in itself. Even Luehrmann agreed that total integration was the ultimate goal and that computer use should eventually infiltrate the subject area courses. The beachhead established in the computer literacy course was intended to be the practical place to start.

If this is the case, are schools who have implemented the beachhead approach moving toward integration? Becker's findings strongly suggest this trend is well under way. While programming and computer literacy remain the predominant use, a significant decrease has occurred over the past two years. On the other hand, the use of tool type software has increased from 10% to 30% over the same period.

For those in a building or district with a firmly entrenched beachhead of hard-earned computer literacy courses, the current trend toward integration introduces a new set of questions and potential problems. First, are the benefits of integrating worth the hassle of changing what took so long to get started? And second, if this is so, what are some ways to make the transition as smooth as possible?

Is Integration Worth the Hassle?

The answer to this question depends upon how the computers are integrated into the classroom.

If the means of the integration is in the use of productivity tool software, the benefits are well worth the effort. Among the benefits are the following:

1. The skills developed in using tool software are applicable in many subject areas as well as in the work place. For example, the ability to use one data base management system is a valuable asset in any of the sciences or social sciences and is a skill they are likely to use later in their work experience.

2. The skills involved in using one productivity tool application are highly transferable to other packages of similar nature. For example, once a student has learned to use one word processor, it is fairly easy for him/her to learn to use a more powerful or a different word processor later.

3. Productivity tools can be much more cost effective over the long run than drills and tutorials. For example, multiple copies of a word processor can be used by more students in more subjects for a longer portion of the school year than a comparable purchase of any CAI product.

4. Some productivity tool software can also be useful to the teacher for test generating, recordkeeping, grades, etc.

5. Productivity tools are an appropriate use of the computer that cannot be duplicated by any other media. Many CAI packages have been criticized because they do not achieve anything that cannot be duplicated just as effectively with a less expensive media such as a workbook or filmstrip. This criticism certainly is not true for word processors, data base managers, etc.

Tips for Making the Transition

The transition from the computer literacy beachhead to total integration can be extremely difficult and will not occur quickly. The suggestions below represent a list of ideas others have found helpful when making the transition. Realizing that the hardware, software, and staffing resources differ greatly between schools, it should be understood that no one school is likely to or should implement all of these suggestions.

1. Revise Computer Literacy Curriculum--Redefine the content of the computer literacy course to focus on keyboarding and word processing at the elementary level, and expand to include data base management, spreadsheets, and graphing utilities at the secondary level. Minimize the programming (including Logo) in the computer literacy course reserving programming for the more serious computer student. The goal is to develop a student population which is comfortable using the most common computer applications.

2. Standardize Carefully Selected Software--Standardization of a selected set of productivity software will simplify staff training, enable buildings or districts to take advantage of multiple unit discounts, and allow for sharing data files containing story starters, templates, data bases, etc. Over time, teachers just getting started with the classroom use of computers will also be more likely to have someone to go to for help.

3. Student Access--Encourage students to use the productivity software for assignments in other classes. Provide as much student access to the hardware and software for this purpose as the schedule will allow.

4. Staff Inservice--Arrange for staff inservice on the productivity tool software used by your building. Provide incentives for teachers to take the classes.

5. Scheduling Hardware Use--Make scheduling of hardware and software for use in subject area classes "hassle free." Nothing will squelch a teacher's enthusiasm faster than encountering a big hassle when trying to acquire the machines for a classroom activity.

6. Computer Assistants--Develop a partial credit course where competent student assistants accompany the computer system into a classroom to set up and assist students with the use of the system.

7. Ongoing Class Projects--Encourage teachers to develop and share ideas for ongoing class projects involving the computer. Such projects can operate in the background and complement a unit of study. One computer can serve as the central collection point for assigned work of many students. Such activities include: team writing projects; class newsletters; designing, building, and using a data base; surveys; and optimization studies, to name a few.

8. Application Library--Develop a library of classroom tested data files, templates, and handouts which can be shared among interested teachers.

9. The Technology Minute--Give a very short presentation at every staff meeting, highlighting a successful classroom application.

10. Peer Modeling--Provide release time for teachers who have developed successful applications to work with other teachers wishing to implement the idea.

11. Forums--Provide a forum where teachers of similar content area can readily share their ideas with other interested teachers.

12. Content Area Curriculum Review--Provide for content area curriculum revision whereby groups of teachers prepare units and/or lessons which make use of productivity tools.

13. Setting Goals--Encourage teachers in the subject areas to make the classroom use of the computer a goal or target area as part of their teacher evaluation.

References

Becker, Henry Jay. "Second National Survey of Instructional Uses of School Computers," Instructional Uses of School Computers, June 1986, pp. 1-11.

THE CONSTRUCTION OF THE COMPUTER COMPETENCE TESTS

Philip M. Linerode

The Psychological Corporation

Abstract

The Computer Competence Tests (CCT) series was constructed to meet the growing assessment needs of computer curriculums in the United States. Based on a comprehensive analysis of computer curriculums across the country, the CCT assesses student knowledge in the areas of the historical impact of computers, computer operations, and computer applications.

A sample of 6,880 students in grades 7 through 12 and adult education was used.

The CCT functions as a criterion-referenced test with user norms available; therefore, the achievement of students may be compared to specific curriculum objectives as well as to the achievement of other students at the same grade level.

Introduction

In Dayton, Ohio, a colleague attending this conference observed that specific tests for assessing computer competencies were lacking. Books were written, and tests were developed to instruct and test students in specific programming language skills. Curriculums were being consolidated that sought to define the scope of the field. Yet, there did not seem to be an instrument, or set of instruments, that would allow assessment with some sense of national perspective. It was toward this goal that effort was directed to develop and make available an instrument that would assess a wide range of computer curriculum content. Given this year's conference theme of lifelong computing, a presentation of such a test seemed appropriate.

To address this concern, a set of five tests, or modules, was developed. The Computer Competence Tests series was created. The tests examine three strands of computers and computing: development and impact of computers, computer operations, and applications of computers.

Overview

The Computer Competence Tests (CCT) series has been designed to meet the growing assessment needs of computer courses in junior high schools, senior high schools, and adult education centers. The CCT is the result of a comprehensive analysis of computer curriculums throughout the United States. This analysis yielded important information about the areas of computer competence which are emphasized in courses and which must be mastered to ensure future scholastic success in computer study. The competence areas tested by the CCT include the historical development and current impact of computers, computer operations, and the application of computer technology.

The CCT is a series of five modules, each of which is a pencil and paper, multiple-choice test to be completed in one class period (approximately 25 minutes). The five modules are:

Module 1 - History and Development
Module 2A - Operations, Level 1
Module 2B - Operations, Level 2
Module 3A - Computer Applications, Level 1
Module 3B - Computer Applications, Level 2

A list of the objectives in each module is found in Appendix A.

The tests are designed primarily for use by junior and senior high school students, although they may be administered to adults when educationally appropriate and to students as young as Grade 4 who are able to read the questions. Any module or combination of modules may be administered to a given class, depending upon the course content and intended use of test results.

Results may be used for a number of purposes. The test can provide placement, formative and summative information. When used as a placement tool, students can be placed in the curriculum at a point that seems best for their current level of knowledge. Since the CCT measures knowledge over twenty-three objectives, each module can be specifically compared to a curriculum, a match can be made, and a student can be placed in an instructional program that best fits. The test can be used to assess work through a course. The test has objective-specific questions. As such, a student's scores can be observed both at a module level and an objective level. This knowledge aids in instructional planning by specifying areas of computer knowledge which have and have not been mastered. If the curriculum match is exact enough to fit a course or set of courses, the instruments can be used as an end-of-session test. The CCT can function as a unit, mid-term, or final exam. Again, scores are objective- and module-specific, which allows for detailed analysis of results.

The applicability of CCT to various curriculums is enhanced by its independence from computer hardware and language. CCT is based on generic operations and applications of computers which comprise essential basic computer competence. Knowledge of any specific type of computer language code or computer model is not tested.

The CCT is a fully supported test with the following pieces:

Directions for Administering;
Test Booklets;
Ready Score™ Answer Documents;
a Class Summary Report; and
a Manual of Technical Issues.

User Group Information

Comparisons across curriculums may be made through consideration of the CCT's user norms, based on the scores of junior and senior high school students who participated in user group testing in April 1985. Table 1 details those students who participated in the pilot testing. The numbers are duplicated counts, allowing for a student to be counted each time a module was taken. In these tables, "Junior High" refers to students in grades 7 and 8, and "Senior High" to those in grades 9 through 12. Fifty-six adult participants took the test. Although that number was too small to provide meaningful results by itself, it is used in the grand totals. Therefore, "All" refers to all test-takers, 7th grade through adult.

Table 1

Number of Students Participating by Module

	Junior High	Senior High	All
Module 1	2105	2098	4232
Module 2A	2735	3028	5819
Module 2B	91	1892	1993
Module 3A	1255	1108	2373
Module 3B	0	500	500

Table 2 lists the unduplicated count of students who participated by grade. Each student is counted only once.

Table 2

Number of Students Participating by Grade

Grade 7	2,033
Grade 8	1,210
Grade 9	685
Grade 10	717
Grade 11	1,032
Grade 12	1,147
Adult	56
All	6,880

Students were tested in 65 districts in 28 states with the following regional breakdown: Northeast - 15 districts; South - 20 districts; North Central - 26 districts; West - 5 districts. The list of schools by region can be found in Appendix B.

Each school chose the module, or modules, to give its students. The number of students in each grade and each module were self-selections, not a pre-assigned number per grade or per module.

Test Development

Items were initially written based on a review of curriculums submitted in response to a national survey.

The items were then reviewed by a panel of experts. A total of 224 items were approved by the review panel. These items were then divided into 28 objectives distributed across the five modules. Once the items were selected, tryout testing was done to select final test content.

At the conclusion of tryout testing, item characteristics were calculated. Final item selection was based on the objective content of the item, the item p-value (the percent of students who answered the item correctly), the point-biserial correlation (the relationship between how a student did on the item compared to the entire test), and a distractor analysis (to what extent and in what pattern the wrong answers were chosen). Based on these criteria, twenty-three (23) objectives using one hundred fifteen (115) items were selected for final CCT use.

The ranges of p-values and point-biserials per module are provided below.

Table 3

P-Value Ranges and Median Value by Module

	P-Value Range	Median P-Value
Module 1	38-85	58.0
Module 2A	31-91	63.0
Module 2B	22-63	49.5
Module 3A	31-86	60.0
Module 3B	21-85	56.0

Table 4

Point-Biserial Ranges and Median Value by Module

	Point-Biserial Range	Median Point-Biserial
Module 1	0.29 - 0.52	0.420
Module 2A	0.36 - 0.57	0.460
Module 2B	0.28 - 0.48	0.400
Module 3A	0.24 - 0.56	0.450
Module 3B	0.22 - 0.61	0.505

As the test was developed, three themes of computers and their use became evident. These themes were a) the history, development, and impact of computers, b) how computers work, and c) how computers are used. The last two themes (operations and applications) covered a wide range of levels, and each of these themes was split into basic concepts and more advanced concepts. Therefore, the test battery was split into five separate tests, or modules. Modules 2A and 2B split the theme of computer operations, while 3A and 3B cover applications.

An examination of the number of students participating by module and the median p-value scores gave support for dividing operations and applications into two levels each. Table 1 showed that Level 1 modules (1, 2A, and 3A) had approximately the same number of junior high students taking the tests as senior high students. Level 2 modules (2B and 3B) had considerably more senior high students taking them. This suggested that the basic concepts tested in Level 1 were taught at both the junior high and senior high levels. However, the more advanced concepts tested at Level 2 appeared to be tested at the curricular level of more advanced high school course work.

The difference in levels was also suggested by the median p-value scores. Although no student took all five modules, the modules with the highest p-values were the Level 1 modules. Level 2 modules had the two lowest median p-values. This suggested that Level 2 items were harder, and thus those modules more difficult.

Another pattern was observed when the module raw score percentile equivalents were compared. In each of the modules, (except Module 3B which no junior high student took) the same raw score yielded a higher percentile score for the junior high student than for the senior high student. This meant that more senior high students received higher raw scores across the module. For example, in Module 1, if a junior high and a senior high student each got 19 items right, the junior high student would have a percentile score of 90. The senior high percentile score would be 75. This means that 90% of the junior high population scored at or below 19 correct, while only 75% of the senior high population scored at or below 19 correct. In all cases the senior high percentile was lower than the junior high group. This is what should be expected.

A similar pattern of performance was observed when the score groups for each objective were examined. Even with the more coarse score group clusters used at the objective level, a senior high student must score equal to or higher than a junior high student to be in the same score group. High, Middle, and Low Score Groups were used to designate a student's score at the objective level. The Score Groups were based on a stanine score for each raw score point of each objective. The High score group corresponded to stanine scores of 7, 8, or 9. This group accounted for the top 23% of the scores earned by the total group who took the test. The Low score group had stanine scores of 1, 2, or 3, and represented the bottom 23% of the distribution of scores. The Middle group, with stanine scores of 4, 5, or 6, accounted for 54% of the distribution.

The module numbers, their names, the number of objectives, and the number of items per module are listed in Table 5.

Table 5

Test Content Format

Module	Name of Module	# of Obj.	# of Items
1	History and Development	5	25
2A	Operations, Level 1	5	25
2B	Operations, Level 2	4	20
3A	Computer Applications, Level 1	5	25
3B	Computer Applications, Level 2	4	20

Reliability

Reliability was determined using measures of internal consistency. In this case the Kuder-Richardson #20 (KR 20) statistic was calculated for each module. A high KR 20 correlation indicates that the test has homogeneous test items. This suggests that a student would score about the same on any given sample of the test items.

Table 6 provides the Kuder-Richardson #20 reliability coefficients for the five modules. Reliabilities range from 0.73 to 0.85.

Table 6

Kuder-Richardson #20 Reliability Values by Module

Module	Kuder-Richardson #20
Module 1	0.80
Module 2A	0.85
Module 2B	0.73
Module 3A	0.82
Module 3B	0.82

A second measure used to judge the reliability of scores was the Standard Error of Measurement. The Standard Error of Measurement (SEM) provided an estimation of the error range of any student's test score in reference to that student's "true score". The SEMs for each module of the Computer Competence Tests are listed in Table 7.

Table 7

Standard Error of Measurement by Module

Module	SEM
Module 1	2.17
Module 2A	2.05
Module 2B	2.04
Module 3A	2.10
Module 3B	1.86

User Norms

Percentile conversions used at the module level are based on the group described in the User Group Information section. They are called user-group norms because a nationally normed group was not sought as with a major standardized achievement test. There were a number of reasons for this. The schools participating were self-selecting; that is, they were offered all modules and chose the one or ones they wanted to use. The criterion-referenced focus of the test and the number of students needed for reliable item statistics also did not require a national sample. However, the major reason for the lack of a statistically representative national sample is the diversity of computer curriculums. There are major differences in the curriculums, both between schools and across grades. For example, in one school a specific course in computers was offered, while in another, computer instruction was an addition to a math class. Finally, a balance of sex, regional representation, ethnic diversity, and socioeconomic levels was not sought and did not occur.

The user group is provided for reference, but should not be thought of as a nationally representative sample. Over time you may wish to compile your own building or district norms. These may well prove more meaningful to you.

Module Percentile Conversions

The percentile conversions are made at the module level and are based on the group described in the previous section. Since these are user-group norms, the balance of regional representation, ethnic diversity and socioeconomic levels is not controlled as for a standardized achievement test.

Percentile scores are provided for junior high (Grdes 7-8), senior high (Grades 9-12), and all students combined. Scores exist for each module. There is a user-group percentile score for each raw score. In the example below, a senior high student got 19 items correct on Module 1. The corresponding percentile score for high school students is 75. This means that this student scored equal to or better than 75 percent of the other senior high students who took Module 1. The actual tables are in the CCT Manual, pages 6-8.

Module Percentile Comparison (Example)

Module 1
Senior High (9-12)

Number Correct	Module 1
25	99
24	98
23	95
22	90
21	85
20	80
19	75
18	70
-	-
-	-
1	5
0	1

Score Group by Objective

Score groups are divided on the basis of stanine scores and are based on the user group described earlier.

Score groups are provided for each objective within each module. The score groups are further divided by junior high (7-8), senior high (9-12), and all students combined. For each objective, a percentile score was obtained. That score was converted into a stanine, which was converted into a "HIGH", "MIDDLE", or "LOW" score group.

"HIGH" means that a student who received that score was in the High group. That student's stanine score would be a 7, 8, or 9 and be placed in approximately the top 23% of the total group who took that objective.

"MIDDLE" means that a student who received that score scored somewhere in the middle of the pilot group's distribution. This is a stanine score of 4, 5, or 6.

"LOW" means that a student who received that score scored in the Low score group of the user group for that objective. This is a stanine score of 1, 2, or 3.

In the example below, a junior high student got 4 correct on Objective 1 of Module 1. The corresponding score group for junior high students is "MIDDLE". This means that this student scored in the middle 54 percent of junior high students who took Objective 1 of Module 1. The actual tables are in the CCT Manual, pages 10-14.

Score Group by Objective (Example)

Module 1
Objective 1
Junior High (7-8)

Number Correct	Score Group
5	HIGH
4	MIDDLE
3	MIDDLE
2	LOW
1	LOW
0	LOW

82

Appendix A

Module 1: Development and Impact

Objective 1: Evaluate career modifications produced by the computer revolution.

Objective 2: Recognize typical computer applications in areas of business, personal use, government, and education.

Objective 3: Describe various generations of computers: identify trends, general time frames, hardware improvements.

Objective 4: Understand the moral and ethical issues presented by the computer revolution: computer crime, piracy, copyright, security issues.

Objective 5: Recognize vocabulary and basic concepts associated with computer usage.

Module 2A: Computer Operations - Level 1

Objective 1: Read and interpret a simple decision table.

Objective 2: Read and interpret logic flow diagrams.

Objective 3: Identify the various hardware structures associated with the input, processing, memory, and output components.

Objective 4: Recognize basic steps involved in creating a computer program, including the concept of documentation.

Objective 5: Describe the functions and parts of a computer system: input, processing, memory, output.

Module 2B: Computer Operations - Level 2

Objective 1: Identify design procedures in creating computer programs.

Objective 2: Identify various logical patterns: simple sequence, decision (selection), loop, branching.

Objective 3: Describe types of program errors and debugging procedures.

Objective 4: Understand the uses of various number systems involved in computer operation: binary, decimal, hexadecimal.

Module 3A: Applications - Level 1

Objective 1: Describe capabilities and limitations of the computer.

Objective 2: Differentiate the types of computer applications.

Objective 3: Understand basic principles for maintenance of computer hardware.

Objective 4: Identify major types of user software.

Objective 5: Recognize the major uses of popular programming languages: BASIC, FORTRAN, COBOL, Pascal, LOGO, LISP.

Module 3B: Applications - Level 2

Objective 1: Evaluate software needs and recommend an appropriate software package.

Objective 2: Use special key functions from packaged software.

Objective 3: Use system commands from packaged software.

Objective 4: Identify major components of programming languages: types of statements, files, variables.

Appendix B

List of Participating Schools by Region

Northeast - 15

Middle Atlantic - 9
- Tenafly H.S., Tenafly, NJ
- Clinton Central School, Clinton, NY
- William Floyd H.S., Mastic Beach, NY
- Carlisle Sr. H.S., Carlisle, PA
- General McLane H.S., Edinboro, PA
- Cedar Cliff H.S., Lemoyne, PA
- Red Land H.S., Lemoyne, PA
- West Shore School District, Lemoyne, PA
- Governor Mifflin H.S., Shillington, PA

New England - 6
- East Ridge Middle School, Ridgefield, CT
- Lewiston H.S., Lewiston, ME
- Quabbin Regional H.S., Barre, MA
- Somerville H.S., Somerville, MA
- Pentucket Regional H.S., West Newbury, MA
- Richford Jr.-Sr. H.S., Richford, VT

South - 35

West South Central - 25
- Fayetteville West Campus, Fayetteville, AR
- Central Jr. H.S., North Little Rock, AR
- Lakewood Jr. H.S., North Little Rock, AR
- Northeast H.S., North Little Rock, AR
- Ole Main H.S., North Little Rock, AR
- Ridgeroad Jr. H.S., North Little Rock, AR
- Rose City Jr. H.S., North Little Rock, AR
- Alluwe School, Alluwe, OK
- Leedey Public School, Leedey, OK
- Calallen Middle School, Corpus Christi, TX
- Hemphill Public School, Hemphill, TX
- Atkins Jr. H.S., Lubbock, TX
- Estacado H.S., Lubbock, TX
- Evans Jr. H.S., Lubbock, TX
- Hutchinson Jr. H.S., Lubbock, TX
- Mackenzie Jr. H.S., Lubbock, TX
- Slaton Jr. H.S., Lubbock, TX
- Wilson Jr. H.S., Lubbock, TX
- Brazos Jr. H.S., Orchard, TX
- Prairie Lea School, Prairie Lea, TX
- Central H.S., San Angelo, TX
- Spurger Elementary School, Spurger, TX
- Brazos H.S., Wallis, TX
- Winona H.S., Winona, TX

East South Central - 5
- Bowling Green State Vocational Technical School, Bowling Green, KY
- Lloyd Memorial H.S., Erlanger, KY
- Olive Branch Middle School, Olive Branch, MS
- Maryville H.S., Maryville, TN
- Maryville Middle School, Maryville, TN

South Atlantic - 5
- LaBelle H.S., LaBelle, FL
- SGA H.S., Sardis, GA
- Waynesboro H.S., Waynesboro, GA
- East Rowan Sr. H.S., Salisbury, NC
- Beddingfield H.S., Wilson, NC

North Central - 29

West North Central - 16
- Victoria Grade School, Victoria, KS
- Victoria H.S., Victoria, KS
- Bloomington Public Schools, Bloomington, MN
- Oak Grove Jr. H.S., Bloomington, MN
- Olson Jr. H.S., Bloomington, MN
- Community H.S., Little Falls, MN
- Luverne H.S., Luverne, MN
- Melrose Sr. H.S., Melrose, MN
- Raymond Public School, Raymond, MN
- Waterville - Elysian H.S., Waterville, MN
- Battle Creek H.S., Battle Creek, NE
- Millard South H.S., Papillion, NE
- Plainview H.S., Plainview, NE
- St. Paul Public School, St. Paul, NE
- Groton Central H.S., Groton, SD
- Howard H.S., Howard, SD

East North Central - 13
- Athens H.S., Athens, IL
- East Dubuque H.S., East Dubuque, IL
- Churchville Jr. H.S., Elmhurst, IL
- York Community H.S., Elmhurst, IL
- Batesville Middle School, Batesville, IN
- Albion Sr. H.S., Albion, MI
- Washington Gardner Jr. H.S., Albion, MI
- Carson City - Creptal H.S., Carson City, MI
- Farwell Area H.S., Farwell, MI
- Bellefontaine H.S., Bellefontaine, OH
- Siren Consolidated School, Siren, WI
- Spring Valley H.S., Spring Valley, WI
- Big Foot H.S., Walworth, WI

West - 7

Mountain - 3
- Holbrook H.S., Holbrook, AZ
- Marsh Valley H.S., Arimo, ID
- Twin Bridges H.S., Twin Bridges, MT

Pacific - 4
- Carmel H.S., Carmel, CA
- Carmel Middle School, Carmel, CA
- Raymond Cree Middle School, Palm Springs, CA
- Santana H.S., Santee, CA

CAMPUS SUPPORT FACILITIES

George H. Culp
Michael J. Jacobson, Martha H. Weller
Timothy J. Foley
Richard E. Swanson, Gary L. Lorenzen
Thomas E. Gist, Marilyn K. McQuade, Robert G. Fuller

ABSTRACT: Project QUEST: Innovative Uses of Microcomputers in Teaching and Research

George H. Culp, COM-1, University of Texas at Austin, Austin, Texas 78712

Project QUEST is a long-term effort of the University of Texas at Austin to increase the innovative uses of microcomputers in teaching and research. It is supported, in part, by gifts of microcomputer hardware and software from the IBM Corporation and Apple Computer, Inc.

Beginning in May 1984, faculty members were invited to submit proposals to the project describing new approaches for microcomputer use. These proposals were then reviewed by a 10-member interdisciplinary faculty committee. To date, more than 590 proposals have been received. Almost 400 of these have been approved as projects and awarded hardware and software by the committee. Seventy percent of these projects are directly related to innovative uses of microcomputers in teaching. Approximately 50 percent of the projects are developing new software materials. Overall, an estimated 750 faculty members, 1,800 graduate students, and 6,000 undergraduate students are using more than 1,350 microcomputer workstations provided through Project QUEST. Upcoming activities will test the validity of developed software in larger laboratory and classroom settings, including microcomputer workstation facilities for student use in the General Libraries.

ABSTRACT: Changing Attitudes Toward Computing: A Profile of the Humanities Faculty at the University of Illinois at Urbana-Champaign

Michael J. Jacobson, Teaching Assistant, College of Education and Martha H. Weller, Ph.D., Research Associate, Institute of Aviation, University of Illinois at Urbana-Champaign

The School of Humanities faculty at the University of Illinois at Urbana-Champaign (UIUC) were surveyed in October 1986 to assess their current use of and attitudes toward educational computing. The respondents were generally self-trained in computer use, indicated positive attitudes toward computers, and made frequent use of them. Frequency of computer use, level of general computing skills, computer interest, and anxiety were analyzed according to respondent rank, sex and age. Faculty perceptions of obstacles to computer use in the humanities indicated a need to address issues of funding for hardware, quality of software, training, and technical support. The main faculty interests in applications software included word processing, desktop publishing, graphics, data base management, communications, and computer assisted instruction. A follow-up survey (administered in April 1987) assessed changes in faculty use of and attitudes toward computing over the 1986-87 academic year. While recognizing that humanities faculty do not have the same level of involvement in computing as faculty in more "technical" disciplines, UIUC humanists, as a group, are clearly not "computer-phobes."

ABSTRACT: The Design, Development, and Implementation of a Microcomputer Based Campus Automation System

Timothy J. Foley, Lehigh University, Computing Center, Bethlehem, Pennsylvania 18015

The campus automation system being implemented at Lehigh University provides such office automation activities as electronic mail, text processing, data base access, calendaring, file transfer, and external network access.

The major components of the system are: a digital PBX, with 4,000 data connections between classrooms, dorms and offices; more than 2,000 microcomputers, with 200 distributed among public locations such as the libraries, classrooms and computing clusters, and 400 distributed among individual faculty members; a Network Server (IBM 4381), which is used as a communications hub for the entire campus; and, a microcomputer based Network Access Facility.

The Network Access Facility will provide the following functions: automatic log-in to a variety of mainframes and minicomputers; text and graphics terminal emulation; background sending and receiving of electronic mail; background transferring of files to all supported mainframes, minis, and micros; access to the libraries' on-line data bases; access to external data networks; and high quality printing of documents through the Network Server.

The Network Access Facility and Network Server applications are being developed as part of a joint project with IBM and the McGill University MUSIC Product Group. The major objective of this project is to provide the Lehigh campus--with more than 7,000 students, faculty and staff--with computer networking services that are easy to use and accessible from nearly every location on campus.

ABSTRACT: Instructor Workstations and Their Classroom Effectiveness[1]

Richard E. Swanson, Gary L. Lorenzen, Thomas E. Gist, Marilyn K. McQuade, Robert G. Fuller,[2] Department of Physics, United State Air Force Academy, Colorado Springs, Colorado 80840

The United States Air Force Academy Department of Physics is engaged in research to develop computer-controlled Instructor Workstations as aids to the instructor in the classroom and to determine their effectiveness. The workstation we are developing is controlled by an IBM PC-AT compatible Zenith Z-248 microcomputer and includes a videodisc system with graphics overlay, an analog data acquisition system, hardcopy output, a Unix-based central file server,[3] and associated educational support hardware and software. The workstations were designed to support in-class applications of technology such as: videodisc presentation, lesson presentation and simulations, demonstrations and laboratories, drill and practice, and course administration. We developed two prototype workstations and used them in introductory physics classes during the spring 1987 semester. We evaluated the performance and attitudes of students who were exposed to lesson presentation using the workstations, and surveyed instructors to help determine the usefulness of these workstations as in-class teaching tools. We will present a description of the workstation and our project and discuss the results of our effectiveness studies.

[1] Work supported by the Advanced Training System Systems Program Office, Aerospace Medical Division, Air Force Systems Command.

[2] Distinguished Visiting Professor on leave from the University of Nebraska-Lincoln.

[3] An AT&T 3B5 provided by AT&T Information Systems.

COMPUTERS AND THE TEACHING OF LITERATURE

Stephen Marcus, Ph.D.
South Coast Writing Project
University of California
Santa Barbara, CA

ABSTRACT

Applications of technology to teaching literature run the gamut from the simple to the sophisticated, from specialized word processing to hypertext media. This presentation will begin with a brief look at the state of the art and future directions for such developments.

The focus of the presentation, however, will be on two related approaches: student-constructed data bases and student-authored interactive fiction. In both cases, students can correlate "alternative itineraries. . .within the charted critical landscape" (Paul Hernadi).

Data base software provides a specialized but flexible tool to help students think, talk, and write about literature. Properly used, it can (1) establish a collaborative learning environment, (2) demonstrate a powerful information-handling tool, (3) teach inquiry skills, and (4) put students in charge of the technology by allowing them to construct data base structures that meet their own research needs.

Similarly, students can write (not just read) interactive fiction, modeled on adventure games. Such software helps students develop their skills as information artists and knowledge architects, establishing a different sort of "data base" as they study and write about literature. Traditional analytical perspectives (like character, theme, point of view, and audience analysis) may be explored in unusual and inventive ways by allowing students to apply their insights to the production of literary landscapes. Students are able to "reintroduce the element of play into criticism" (Hernadi).

Sherry Turkle at MIT has characterized the computer as "an evocative object, an object that fascinates, disturbs equanimity, and precipitates thought" (*The Second Self*). British author Daniel Chandler has noted that "the legacy of literature. . .has been a 'civil defense' strategy by English teachers to protect children from the evils of modern industrial society. . . . For such teachers the computer is often just another invasive influence in the classroom. It will no doubt be many years before there is any general shift in this stance" (*Computers and Literacy*).

We can certainly all stand to rethink our notions of how, and with what consequences, computer technology can affect the ways students think, talk, and write about literature. We should constantly reassess the problems and promise of technology. We should make explicit what we know and don't want to know about computers and the teaching of literature.

A VIDEOTAPE SURVEY OF TECHNOLOGICAL ADVANCES IN MILITARY TRAINING

J. D. Fletcher
Institute for Defense Analyses
Washington, DC

ABSTRACT

For almost 30 years, the Department of Defense has been a major investor in research, development, testing and evaluation of computer applications in training. This investment has produced and continues to produce significant innovations that make their way into public education. This presentation surveys some of the more advanced computer applications now being developed for military training. These applications include videodisc techniques, computer speech generation and recognition, artificial intelligence techniques, simulation networking, and computer generated animation. Where data are available, the costs and effectiveness of these techniques are discussed, and where videotapes are available, the applications are demonstrated using this medium. Over a variety of subject matter and in a number of settings, it is found that computer applications in training generally reduce student time to reach criterion by about 30 percent and that they generally cost about 28 cents less per unit of effectiveness than do more conventional approaches to instruction.

AN INTRODUCTORY PROBLEM SOLVING AND DESIGN COURSE

Shirley J. Beil
Normandale Community College
9700 France Avenue South
Bloomington, MN 55431

Abstract

The following paper describes an introductory level Computer Science course in problem solving. The paper describes the reasons for implementing the course, its content, and the reactions of the students. Several design tools used in the course including break-out diagrams, data space diagrams, and complete traces are discussed. The choice of the APPLE Logo language and that of Karel The Robot (Pattis 1981) for the course is discussed and the languages' roles in the course is explained.

Background

A one quarter, 5 credit course in beginning problem solving was introduced at Normandale Community College, Bloomington, Minnesota in Fall 1984. The enthusiasm of the over 1000 students who have now taken this course has encouraged me to submit this paper. Figure 1 below shows the distribution of grades for the 13 sections (350 students) which I taught between Fall 1984 and Summer 1986:

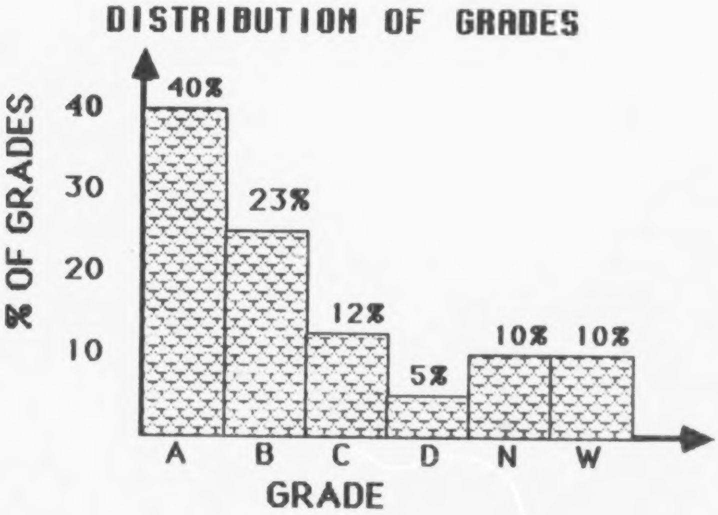


Figure 1 - Distribution of grades

I have a strong interest in structured programming and a belief that a student's introduction to problem solving and design should be a structured one. Prior to proposing this course my experience included teaching introductory courses in BASIC, FORTRAN, and Pascal. I was frustrated by:

a) having to cover so much of the same material in each course

b) finding it difficult to get into sophisticated topics in a one quarter course

c) difficulty in introducing structure in a language whose syntax is not inherently structured

d) lack of a common background and vocabulary of students in these courses

Problem Solving and Design is now a prerequisite to Pascal and FORTRAN. Its effect has been to allow significantly more material to be covered in both courses. More importantly, I have found in teaching Pascal that an entirely different and more coherent presentation of the material is possible because of the vocabulary and skills which the students now bring to the course. The acceptance of the course by CSci instructors has been so enthusiastic that starting Fall 1987, the Problem Solving and Design course will be a prerequisite to a new one quarter, 5 credit BASIC course.

Content

In designing the course we wanted to introduce students to:

a) top-down design

b) at least one structured language which allows passing of parameters

c) at least one block-structured language

d) the VAX and APPLE computers

We did not wish to teach the syntax of any language which might be taught in a subsequent course. We thus chose both the APPLE Logo language and that of Karel The Robot (Pattis 1981) to be those introduced in the course.

Our course starts with a one week introduction to top-down design. Algorithms are represented using break-out diagrams composed of modules written in flow block (Nassi-Schneiderman) form. Primitives, subform calls, decisions, and loops are the constructs used to build each module. These are placed in sequence or nested as appropriate to complete the module.

We have chosen to introduce the material on top-down design using the concept of the robot called Karel. Karel's control structures include single- and double-alternative decision structures, ITERATE loops and WHILE loops. Appendix A shows the features of Karel. A break-out diagram is illustrated in Figure 2.

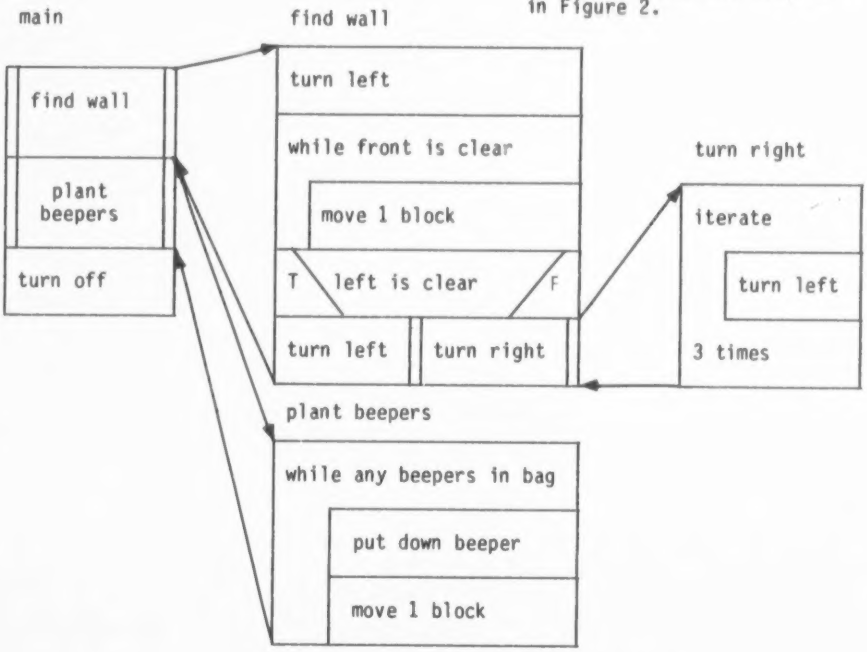


Figure 2 - Karel break-out diagram

The simulation of Karel on our VAX 11/750 computer (also available for other hardware) is the first computer contact our students experience. Many important topics including the file management and editing systems on the VAX are introduced quickly by using the Karel simulator.

Once students are comfortable simulating robot programs which they have been given, the syntax of Karel, which is a block-structured language similar to that of Pascal, is introduced using the syntax diagram shown in Appendix B. Students learn to write the Karel code for the top-down designs which they completed earlier in the quarter. Students must analyze the structure of each module in turn to determine whether each is one instruction or a sequence of instructions. This is necessary in order to determine the placement of BEGIN/END pairs and is valuable in helping the students gain a better understanding of the algorithm. Students are highly motivated to complete their assignments by the fact that they will actually be able to see Karel accomplish his task when the code is submitted to the simulator.

We then move away from Karel and into a more general problem solving environment - one which allows us to manipulate variables. We discuss generalizing, extending and embedding algorithms and we teach the students to represent algorithms using pseudocode and flow charts, as well as flow block diagrams. We spend significant time having the students change the representation of an algorithm given in any of the above three forms into the other two forms. This has been found to be time well spent.

Next we teach students how to do a complete trace of an algorithm, with emphasis on the dynamics of the algorithm. A complete trace shows the following:

a) every value that every variable takes on

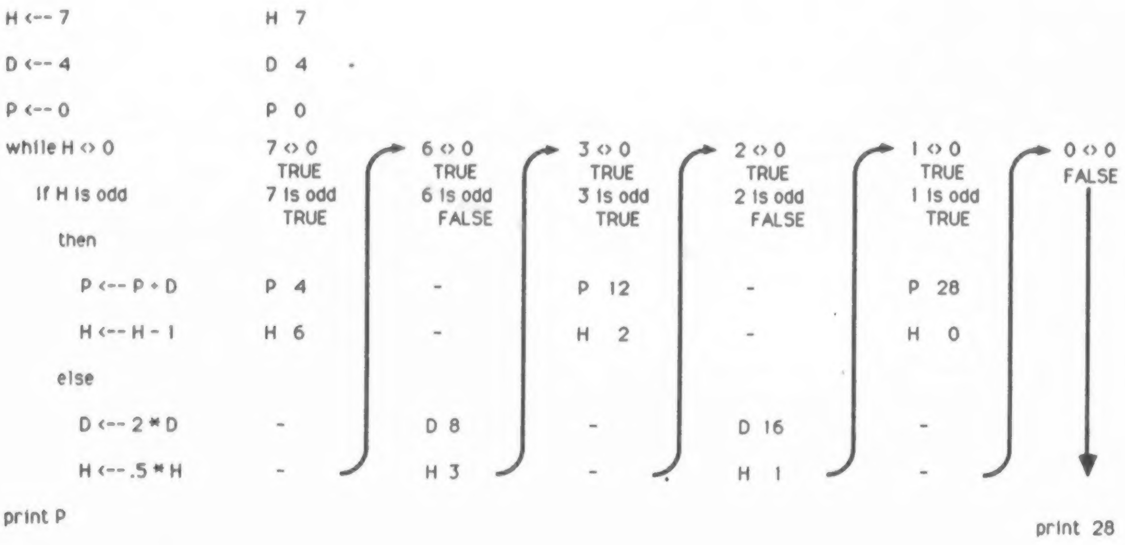
b) every test

c) the truth value of every test

d) every item printed

e) the flow of control

A complete trace of the Peasant Pebble Product problem (Motil 1984) is found in Figure 3. Students are able to analyze quite complex nested logic using this type of trace.



H <-- 7	H 7					
D <-- 4	D 4					
P <-- 0	P 0					
while H <> 0	7 <> 0 TRUE	6 <> 0 TRUE	3 <> 0 TRUE	2 <> 0 TRUE	1 <> 0 TRUE	0 <> 0 FALSE
if H is odd	7 is odd TRUE	6 is odd FALSE	3 is odd TRUE	2 is odd FALSE	1 is odd TRUE	
then						
P <-- P + D	P 4	-	P 12	-	P 28	
H <-- H - 1	H 6	-	H 2	-	H 0	
else						
D <-- 2 * D	-	D 8	-	D 16	-	
H <-- .5 * H	-	H 3	-	H 1	-	
print P						print 28

Figure 3 - Complete trace of an algorithm in pseudocode form

We then introduce the FOR and UNTIL loops which were not available in the Karel language and give students a chance to practice designing algorithms which parallel the verbal description of problems.

Next we use APPLE Logo as the vehicle to introduce parameter passing from one module to another in a top-down design. We have included definitions for the WHILE, UNTIL and FOR loops in the startup file. In this way Logo can be experienced as a structured language with all stuctured constructs present. We include discussions of interactive programming, prefix and infix operators, built-in and user-defined functions, logical and relational operations, random numbers, local and global variables, actual and formal parameters, parameter passing and, sometimes, recursion. We make extensive use of data flow and data space diagrams (Figure 4).

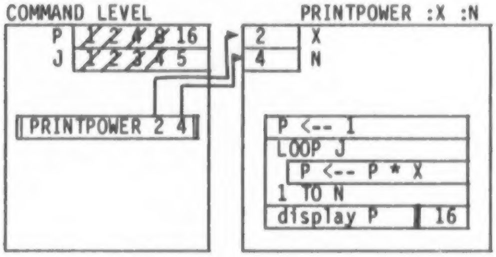


Figure 4 - Data space diagram

Logic is normally covered in the next week. The last major topic is generally a discussion of data structures, mainly one- and two-dimensional arrays.

Future

We have experimented with several required and supplementary text books for the course. Karel The Robot (Pattis 1981) is currently the only text required in all sections. The two Logo references (Billstein 1986 and Martin 1984) contain much more information about Logo than we can cover in our course. Programming Principles (Motil 1984) is an excellent reference but our students have found it quite difficult to read. We have found it necessary to develop on campus a large amount of support material for the course. Currently both students and instructors are reasonably satisfied with the combination of materials used, but the lack of a single text remains a negative factor in the course. We have discussed developing our materials into a text book, but with continual changes in the CSci curriculum, the time to embark on such a major undertaking has not been found. It would be helpful to know the level of interest in such a text by those reading this paper.

The development of this course has been an incredible amount of work. We have received no release time or supplementary pay. On the other hand, I have found the continuing development of the course a personal growth experience and the teaching of the course a very positive experience.

I will close with some student comments:

> "I think that this course should be a prerequisite for all programming classes. This class helps to promote a uniform approach to programming. This is needed in the computer world."
>
> "I liked the idea of the various languages discussed. It showed how wide the range of languages is and the importance of knowing at least a little about several languages."
>
> "The course was well planned and presented to be of benefit to students from a variety of computer backgrounds - from beginners to professionals."
>
> "I have really enjoyed this course. It was interesting and fun."
>
> "This course has taught me how to think."

References

Billstein, R., Libeskind, S., & Lott, J. W. Apple Logo Programming and Problem Solving. Menlo Park : Benjamin/Cummings, 1986.

Martin, D., Prata, S., & Paulsen, M. Apple Logo Programming Primer. Indianapolis : H. W. Sams, 1984.

Motil, J. M. Programming Principles An Introduction. Boston : Allyn and Bacon, 1981.

Pattis, R. E. Karel The Robot. New York : John Wiley and Sons, 1981.

Appendix A

Karel is visualized as a wedge shape which is able to face north, south, east, or west and dwells in a world bounded by impenetrable walls on the west and south. In addition his world is crisscrossed by horizontal streets and vertical avenues. There are two other elements in Karel's world. They are beepers and wall sections. Beepers emit a very soft sound. Karel can hear a beeper only when he and the beeper are on the same corner. Wall sections can be constructed between any two corners.

Karel is able to execute 5 "primitive" instructions:

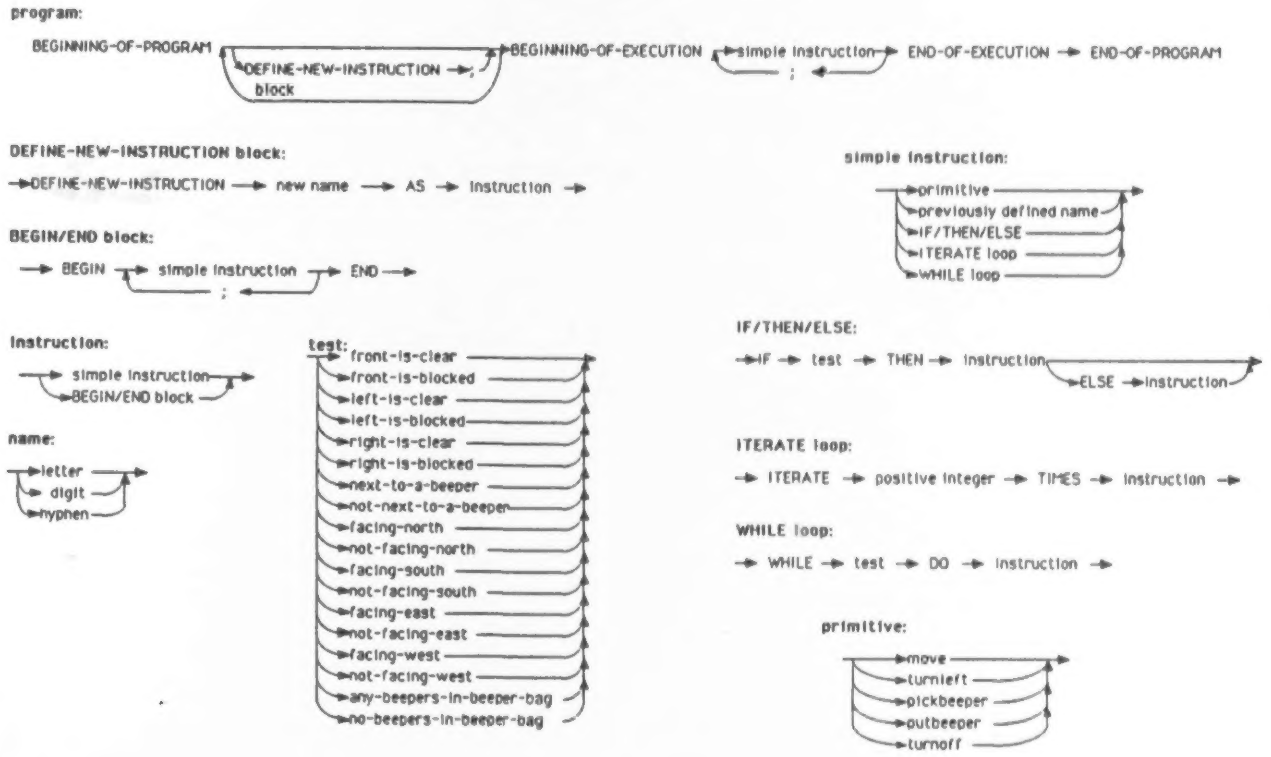
1. move 1 block in the direction in which he is facing;
2. turn left 90 degrees;
3. put down a beeper;
4. pick up a beeper;
5. turn off.

Karel can determine:

1. in which direction he is facing;
2. if there is a wall section 1/2 block to his left, right, or front;
3. if there is a beeper on the corner on which he is located;
4. if he has at least one beeper in his soundproof beeper bag.

APPENDIX B

KAREL THE ROBOT - SYNTAX


program:
BEGINNING-OF-PROGRAM
DEFINE-NEW-INSTRUCTION block
;
BEGINNING-OF-EXECUTION
simple instruction
;
END-OF-EXECUTION
END-OF-PROGRAM
DEFINE-NEW-INSTRUCTION block:
DEFINE-NEW-INSTRUCTION
new name
AS
instruction
BEGIN/END block:
BEGIN
simple instruction
;
END
instruction:
simple instruction
BEGIN/END block
name:
letter
digit
hyphen
test:
front-is-clear
front-is-blocked
left-is-clear
left-is-blocked
right-is-clear
right-is-blocked
next-to-a-beeper
not-next-to-a-beeper
facing-north
not-facing-north
facing-south
not-facing-south
facing-east
not-facing-east
facing-west
not-facing-west
any-beepers-in-beeper-bag
no-beepers-in-beeper-bag
simple instruction:
primitive
previously defined name
IF/THEN/ELSE
ITERATE loop
WHILE loop
IF/THEN/ELSE:
IF
test
THEN
instruction
ELSE
instruction
ITERATE loop:
ITERATE
positive integer
TIMES
instruction
WHILE loop:
WHILE
test
DO
instruction
primitive:
move
turnleft
pickbeeper
putbeeper
turnoff

AN EDUCATOR'S SOLUTION TO COMPUTER REPAIR PROBLEMS

Richard G. Moll
Waterford Schools
Waterford, Michigan 48095

ABSTRACT

The problems of maintaining and repairing microcomputers have been following right behind the explosion of classroom computers across the nation. An educator's solution to these problems is to teach students to clean and fix computer equipment to manufacturer's specifications. This is exactly what Waterford Schools in southeastern Michigan has been doing since early in 1984.

Our program in computer electronics and repair has added a special and unique contribution to our computer science department. It has offered students a way to learn difficult electronic concepts by focusing on a concrete goal which provides meaning and motivation for learning. This program also saves thousands of tax-payer dollars annually. The combined results of accelerated learning and dollar savings make this an idea I hope you find enticing.

The purpose of this paper is to show how this program evolved, give a curriculum outline and provide a positive cost/benefit analysis. It is my hope that this example will encourage others to try innovative and educationally sound curriculum enhancements.

INTRODUCTION

Waterford Schools is a suburban, pre-k to 12 school district of some 12-13,000 students. As our use of microcomputers grew to its current 500+ number we faced the repair and maintenance problems today being felt by schools and businesses alike. While vendors are charging $30-$50 for a checkout/service charge we are opting to repair most if not all of our computers in-house. What is special about our approach is that we did not hire a technician to repair and maintain our computers - we are training STUDENTS to repair our computers to manufacturer's specifications.

Our computer electronics/repair program has proven to be a wonderful opportunity for high school students. In the process the school district also has saved thousands of dollars in repair and maintenance costs annually since 1984. I will give a short description of how this program came into being and then outline the curriculum and economic benefits of what I believe to be an innovative solution to a problem shared by schools across the country.

BACKGROUND

Over a period of about five years the number of computers in use in our district grew rapidly from a few ten's to a several hundred. As the machines passed out of warranty the problems of maintenance and repair loomed clearly on the horizon. Our goal, then, was to provide a means for maintaining, repairing, improving etc. our machines at low cost to the district.

There are several options for meeting this goal. They are:

1. Do nothing - a poor choice.
2. Contract with a repairer to repair and maintain the machines
   - expensive, few local companies.
3. Hire a technician to repair them in-house
   - a possibility but one that had been tried for AV equipment with only marginal success.
4. Train students to repair the computers at the chip level
   - an educationally interesting proposition but initially expensive and perhaps difficult to do.
5. Ship all repair to the local factory distributor
   - the easiest but most expensive because of their tendency to repair at the board level.

A decision was made to use the last two options. Students would be trained to repair microcomputers. If they were unable to do so then the computers should be taken to the manufacturer's outlet. The following timeline shows some of the decisions that were made to implement our plan.

Winter 1983 - the administration decided that we would repair computers in-house and that I would develop and implement a one semester computer electronics course in the winter of 1984 to train students to repair microcomputers, with an emphasis on the TRS-80 Model I and Model III.

August 1983 - the course instructor was sent to Ft. Worth, TX for one week to be trained by Radio Shack in Model III repair. This instruction permitted our district to order parts and manuals from the manufacturer because we now had a certified repairman.

Fall 1983 - a text book for the course was selected and equipment ordered. A course outline was developed. An account with a national parts division of Tandy was opened.

Winter 1984 - the course was taught for the first time and students did minor repairs and maintenance
(ELECTRONICS I prerequisite).
- inventory started.
- control procedures started.
- paid student volunteer began repair center operations.

Summer 1984 - upgrading and maintenance was done by paid students from computer electronics.
- two students and a teacher were sent to a local repair center and received six hours of repair training for the Commodore 64.

Fall 1984 - selected CE (computer electronics) students elect CE credit to repair computers one hour daily (2 students actively involved).
- inventory was enhanced.
- Commodore's referred to vender.
- decision to continue multi-faceted repair approach.

Winter 1985 - course offered second time.

Summer 1985 - a previous year's student helps new students to learn procedures.
- Commodore C-64 and 1541 drives included as machines to be repaired by students.

Fall 1985 - four students from last year's CE class elect to repair district computers.

Winter 1986 - two CE courses offered.

Summer 1986 - 5 students repair, clean, set-up labs.

Fall 1986 - Introductory Computer Electronics added to curriculum and computer repair students given separate repair credit.
- TEN students elect computer repair course.

CURRICULUM

The curriculum now looks like this :

10,11,12	Computer Electronics I - 1 semester, 1 hr/day - algebra and a computer course recommended
10,11,12	Computer Electronics II - 1 semester,1 hr/day - CE I, Electronics II, or Physics required
12	Computer Repair - 1 year, 1 hr/day - CE II and teacher recommendation required.

Computer Electronics I course outline

1. STATIC ELECTRICITY AND ATOMIC THEORY
2. VOLTAGE
   A. Use of VOM
3. CURRENT
4. RESISTANCE
   A. Series
   B. Parallel
5. OHM'S LAW
6. POWER
7. SOLDERING/PROJECT
8. COMPUTER ARCHITECTURE
9. COMPUTER NUMBERING SYSTEMS
10. Z-80
    A. Registers
    B. Instruction set
11. INTRODUCTION TO ASSEMBLY LANGUAGE PROGRAMMING
12. MAGNETISM
    A. Core Storage
    B. Recording Media
13. LOADING AND SENSITIVITY EFFECTS OF METERS
14. BRIDGE NETWORKS
15. CAPACITORS, INDUCTORS
16. USE OF OSCILLOSCOPES
17. DIODES
18. TRANSISTORS/TESTING
19. TRANSFORMERS/POWER SUPPLIES

Computer Electronics II course outline

1. ELECTRICITY REVIEW
   A. D.C. and A.C. CIRCUITS
   B. Use of VOM and OSCILLOSCOPE
2. ELECTRONICS REVIEW
   A. DIODES
   B. TRANSISTORS
3. INTEGRATED CIRCUITS
   A. READING CHIP NUMBERS
   B. TTL and CMOS

- C. PRE-COMPUTER BASICS
  - binary & hex
  - gates and truth tables
  - flow-charting
- D. FLIP-FLOPS
- E. TROUBLESHOOTING
- F. PRINTED CIRCUIT BOARD PROJECT(s)

4. INTRODUCTION TO COMPUTER REPAIR
   - A. STRUCTURE of a computer
     - registers, memory,etc.
   - B. FUNCTIONING of a computer
     - DOS, assembly programming
     - MPU, ports, fetch, execution, etc.
   - C. SERVICING the Model III/4 and C-64
     - CPU, power supply, address decoding, FDC
     - ports, video monitor, drives, keyboard

Computer Repair course description

There are no formal lessons in this course and therefore no course outline. Students are assigned to the repair center throughout the day and often act without direct teacher supervision. Students repair, pick up and deliver computer equipment on "independent study" basis. They are expected to fill out repair orders, file weekly activity reports, and turn in periodic book or periodical reviews.

As a result of these activities and decisions we have repaired over four hundred (400) computers and we have a high-tech program that offers students practical experiences seldom found at the high school level. Some students were employed during the summer to set-up, upgrade and fix computers. We are providing students with a sound educational experience and in return get a valuable human resource.

ECONOMIC RESULTS

First year results

Economically our decisions are also proving to be meaningful. Most repair services from the manufacturer's dealer costs thirty to fifty dollars per half hour for labor. In addition, dealers seldom repair below the board level. Approximate monetary benefits for the first year of repairs accruing to the district are:

Returns	
Savings in dealer labor costs	$3000
Sav. in add'l costs;alig.,upgrd	$2000
Savings in chip-level repair(11)	$1100
sub-total	$6100

Costs	
Teacher training costs	$ (1500)
Initial cost for course matls.	(10,000)
Inventory holding costs	(1000)
Salaries to students	(800)
Special repair equipment	(500)
sub-total	$(13,800)

Net Savings(cost), after 1 year $(7700)

This analysis of costs and savings shows that we can generate, after one year of operation, significant monetary benefits. An approximate payoff period of two to three years is very good for any capital investment. The educational benefits to students are an added plus. Overall, I believe this has been a worth while approach to solving the repair problem.

Current (third) year results

Returns	
Commodore parts savings	$ 613
Commodore labor savings for 84 repairs	4700
Radio Shack parts savings	62
Radio Shack labor savings for 46 repairs	2043
Cleaning Maintenance savings	490
Sub-total	$ 7908
Costs	
Student salaries	(3596)
Net Savings for third year	$ 4312

As these numbers show this program can pay for itself and is continuing to be cost effective after three (3) years of operation.

EVALUATION

There were and are some problems.
- Too few students take the course. We need more students.
- Inventory needs are hard to determine and costly to maintain.
- We often have backlogs that take two to four weeks to clear up. Supply lines from the manufacturer are long (up to four weeks).

We have attacked many of these problems.
- Student interest is growing.

Enrollment by Course and Year

	CE I	CE II	REPAIR
1984-5	-	24	2
1985-6	-	43	4
1986-7	40	32	10

- Other suppliers for certain parts are being used.
- Inventory needs are being refined.
- Many initial one-time costs have been paid.
- Administrative support has been excellent. There were many people involved in the decision making at every major step. Each person, from fellow teacher to central office staff, was continually supportive. That support is essential to any creative effort in the schools is a given.
- Economic results were positive : an unusual event

We have shown that students can achieve highly when offered the proper opportunities. We have maintained our microcomputers at a savings of thousands of dollars to the tax payers. The computer electronics/repair program that we have started is an example that excellence in education is a common occurance in the public schools. We expect our program to continue and welcome your suggestions and questions. We offer this program as a model to anyone who is interested. If you see such a program as meeting your needs I would be willing to help in any way I can.
Contact me at:

Richard G. Moll
Waterford Mott High School
1151 Scott Lake Road
Pontiac, MI. 48054
(313) 674-4134

AN INTERACTIVE PACKAGE FOR TEACHING PASCAL POINTERS AND DYNAMIC VARIABLES

Jeffrey L. Popyack
Department of Mathematics and Computer Science
Drexel University
Philadelphia, PA 19104

ABSTRACT

An interactive package has been developed to provide a mechanism for learning to use pointers and dynamic variables in Pascal. This addresses the great deal of difficulty students exhibit in learning the material in programming courses. The difficulty stems from a communication gap between the traditional static media of instruction and information of dynamic nature. The software helps bridge this gap by providing a dynamic textbook: Using a visual display similar to that used in most texts, the user is allowed to create and manipulate dynamic variables and pointers, with a subset of Pascal.

INTRODUCTION

It has been my experience in teaching Pascal, data structures, and higher-level computer science courses, that students have a disproportionate amount of difficulty with the concepts of pointers and dynamic variables. In the introductory programming courses, many students fail to understand at all. In advanced courses, a shockingly large number of students completely avoid using them, even in situations which beg for their use. Judging from discussions with computer science faculty from other institutions, this seems to be a common occurrence. A primary question then, is "what has gone wrong?"

It would be easy to blame the instructors, for not presenting the material appropriately. On reflection, this is difficult to accept. We should not expect one common area of difficulty to arise repeatedly (even with bright students), unless there is some intrinsic difficulty in learning that area.

It is also easy to fault the textbooks. Students generally resort to their texts for help when they fail to understand the lecturer. It would seem that students have not received the needed support from their texts. With the myriad of Pascal texts on the market, however, it is unlikely that *all* of them provide poor coverage in this area. Rather, it seems that there is an intrinsic difficulty in presenting the material, as well as learning it.

DYNAMIC VS. STATIC STRUCTURE IN THE TEACHING ENVIRONMENT

With this as the hypothesis, let us attempt to identify a reason: the dynamic behavior of these objects differs conceptually from the static structure of the language, static nature of textbooks, and static mode of instruction. In Pascal, all other variables are declared explicitly. Memory locations are assigned names, and the value of any variable may be discovered with a write statement at any time during execution. The entire form of the program, including control structures, is known prior to execution. This is what is meant by the static structure of the language. Conversely, dynamic variables have no names, which makes tracing the values of such variables difficult for a novice. Because a pointer is an address, its value cannot be disclosed by a write statement in the typical Pascal environment. This lack of access restricts the process of learning by experimentation.

Textbooks are necessarily static. With experimentation severely hampered, a student's first recourse is the text. Although a good text will provide many examples, pictures, and exercises, the lack of interactive dialogue inhibits their value.

The mode of instruction need not be static, but frequently is. A typical scenario involves the instructor drawing a structure on the board, then modifying it repeatedly to show the effects of various commands. The result is a messy display of scribbles, erasures, redirected pointers, and the like. A glance at a student's notes reveals a picture nearly identical to that on the board, with no indication of the original structure or intermediate steps. By the time the student refers back to his notes, the value of a live lecturer is lost.

In an attempt to provide some dynamics to the instruction process, I once devoted half a class period to a "pointers workshop." Although the results were satisfying at the time, an overwhelming majority of the students did not choose to put their new knowledge into practice: programming assignments and exam questions were still done without dynamic variables, except where explicitly required. In many cases, the results were sloppy and filled with errors. The students attributed this to being too uncomfortable with dynamic variables to use them without provocation. Although they were able to deal with the theory, the difficulties with experimentation precluded any attempts to apply them in practice.

Breaking the Barrier

With the availability of the Apple Macintosh comes the ability to create software with high-speed processing and graphics power. The potential for computer graphics in describing dynamic processes has been illustrated wonderfully in the movie, "Sorting Out Sorting" (Baecker,1981). The motion picture medium extends far beyond the static medium of textbook illustration. The ability for user interaction allows software to go one step further. I began work on a pointer variables demonstration package, in an effort to create a sort of "dynamic textbook". The goals were to

(1) provide a visual conceptual model similar to that found in most textbooks,

(2) provide a realistic user environment,

(3) allow learning by experimentation, and

(4) allow its use as a teaching tool as well as a learning tool.

At the time the project was begun, very little software, instructional or otherwise, was available for the Macintosh. Design goals similar to those above were reported for "dynamic algorithm animation" software developed by (Brown and Sedgewick,1985), which was available only on expensive Apollo systems. It is now possible to obtain the software described below, and a wealth of other educational software, through the Kinko's Academic Courseware Exchange, which was created largely in recognition of the suitability of the Macintosh for instruction.

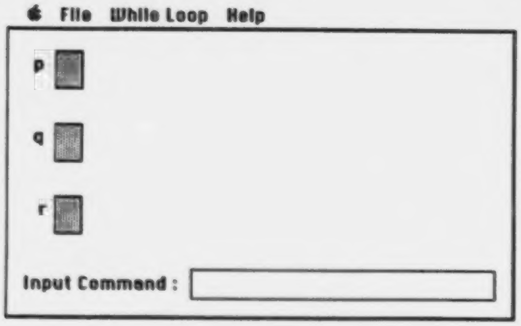
A description of the package that evolved, experience with its use, and a summary are contained in the sections that follow. An explanation of the subject of pointers and dynamic variables can be found in (Popyack,1986), or most introductory Pascal texts.

THE POINTER DEMONSTRATION PACKAGE

A description of the package, and how it addresses the design goals, follows:

Visual Conceptual Model

Upon entering the application, the user is presented with a display depicting the three unassigned variables, p, q, and r. (Unassigned pointer variables are always shown as grey). The prompt **Input Command** and a text box are visible. Any input from the keyboard is displayed in the command box. To execute the command, the user enters a "Return".



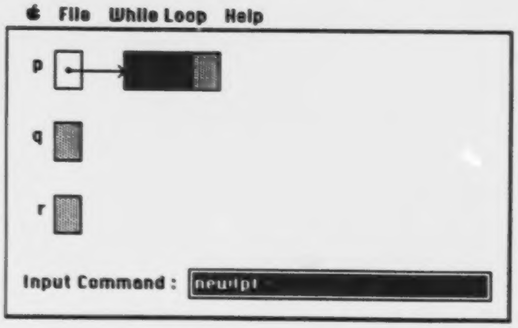
The variables p,q, and r are defined by

```
type
  NodePtr = ^Node ;          Node = record
                               field : char ;
                               next  : NodePtr ;
                             end ;
var
  p,q,r : NodePtr ;
```

The commands available in this Pascal subset are those of the form

new(<NodePtr>)
dispose(<NodePtr>)
<NodePtr> := <NodePtr>
<NodePtr> := **nil**
<Node>^.field := <Node>^.field
<Node>^.field := '<char>'
<Node> := <Node>

For instance, the command **new(p)** causes a new record to be created, with p pointing to it.



new(p)

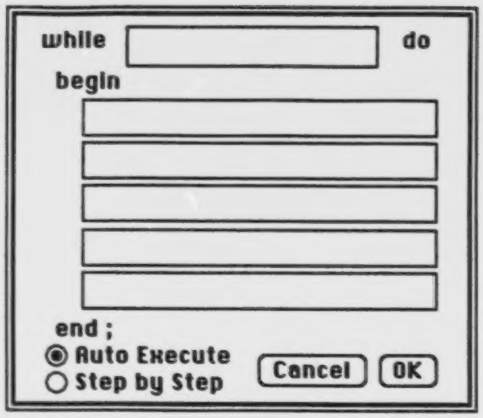
The command **p^.field:='1'** causes the value '1' to appear in the field segment of p↑, and **p^.next:=nil** assigns p↑.next the value **nil**, which is designated on the display by a diagonal line through p↑.next.

User Environment

The interpreter makes every attempt to detect errors, and issue diagnostics, without crashing. If there are syntax errors in a command, an error message is displayed. If there are no syntax errors, but the command refers to an invalid variable (such as p↑.next, when p is **nil** or unassigned), an error message will also be displayed.

In order to illustrate the effects of space limitation (and thus, the motivation for the **dispose** command), only twenty-four memory locations are available. This coincides with the visible area of the screen. It is easy to run out of memory, therefore, if one is not careful.

In order to traverse linked lists, a looping control structure is most desirable. For this reason, the Pointer Demo package contains a while-loop template.



while ☐ do

begin

end ;

● Auto Execute

○ Step by Step

The user may enter a loop condition, and assignment statements which comprise the loop body. Any of the legal Pascal commands discussed previously may be used in the loop body; the loop condition may be virtually any legal Pascal condition, from the same Pascal subset. The user may also select whether to execute the loop *automatically* or *step by step*.

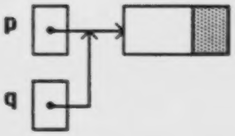
When satisfied with the loop setup, the user selects **OK**, and the statements are checked for syntactic correctness until an error is found, or all are deemed correct. If an error is found, it is identified immediately.

The user decides when to begin loop execution. If **Auto Execute** has been selected, the user sees instant manipulation of the data structure which faithfully reproduces the effects of the loop. If any run-time errors are detected, such as *attempt to access a nonexisting or unitialized variable, attempt to create a new variable when no space exists,* or *infinite loop,* an error message appears and looping is terminated.

If the user has chosen **Step By Step** execution, the message **Hit Return Key** appears in place of the **Input Command** prompt, and the loop condition appears in the command box. Striking the return key causes the condition to be evaluated. If the condition is false, the text "--DONE--" appears in the command box. Otherwise, each subsequent strike of the return key causes the currently displayed command (or condition) to be executed, and the next command to be displayed in the command box. At the end of the loop body, the condition again appears in the command box. This procedure is repeated until either an error occurs or the condition evaluates as false. In either case, the loop is terminated, and the **Input Command** prompt reappears beside the command box.

The rules for implementation of **dispose** are not defined clearly in the Pascal standard, and because of this, an ambiguity is present in the following situation:

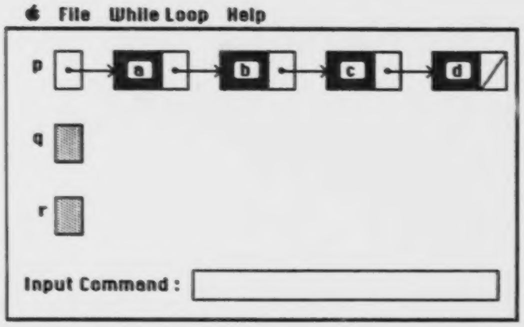
```
new(p) ;
q := p ;
dispose(p) ;
```



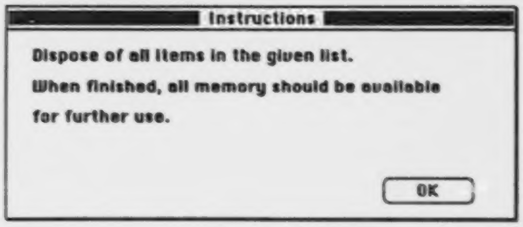
According to the description of **dispose**, this should cause the variable q↑ (which also is p↑) to be destroyed. Yet, this leaves p↑ unspecified. Because the implementation of **dispose** is a design issue, different systems handle this situation in quite different ways. The approach taken by the Pointer Demo Package is that taken by MacPascal -- a variable may only be **disposed** if it is pointed to by exactly one variable. If q↑ exists, and is not pointed to by any other pointer variable, **dispose(q)** causes q↑ to be removed, and q becomes *unassigned.* If more than one variable points to q↑, an error message is displayed.

Learning by Experimentation

A small set of "lessons" is supplied with the package. Each lesson consists of an initial configuration and a set of instructions. The user is asked to manipulate the configuration to produce some other structure. The user is free to attempt any commands to manipulate the structure, and is always able to restart the lesson if necessary.



Input Command : ☐



Instructions

Dispose of all items in the given list.

When finished, all memory should be available for further use.

Use as a Teaching Tool

The instructor may use the package for classroom demonstration, or to create lessons for student use. The user creates an initial configuration by entering commands in the usual fashion. All commands entered while in "Create Mode" are saved, except for those that result in an error message. The user continues entering commands until the initial structure is complete. After entering a brief set of instructions, the user may either choose to save or cancel the lesson. (Cancelling may be done at any time during the process of creating a lesson). This generality allows the package to be used as a teaching tool tailored to the preferences of individual instructors.

EXPERIENCE

Preliminary versions of the package have been demonstrated at the 1985 MacWorld Exposition and MacAdemia '86, as well as at Drexel University. The response has been most favorable, generally accompanied by statements such as "I wish it had been around when I was learning this < stuff >." It has been distributed at Drexel University since spring 1986, and is now available through the Kinko's Academic Courseware Exchange. At Drexel, many students have reported an understanding of the material that would not have resulted from their normal classroom experience. An attempt is underway to measure this understanding. The most gratifying response has come from the Software Development Group, which hires student co-ops for six-month periods to develop Macintosh software. The package has been used as part of the training procedure to indoctrinate students into pointers, whose use is mandatory in systems development. At this writing, it is too early to report on its use at other institutions.

SUMMARY

A recurrent problem in computer science education is the difficulty of communicating dynamic information through static media. In an effort to provide a teaching/learning mechanism for a particular area, that of dynamic variables and pointers, a software package was created which allows the user to enter into a dialog with the computer. The major components of the package are the interpreter, graphic display, and input/output facility, which allows customized 'lessons' to be created and used. The software serves as a "dynamic textbook," providing a component that is missing for this area in the traditional instructor / textbook / computational experimentation approach to teaching. This approach is similar in spirit to the "algorithm animation" work reported recently in (Brown and Sedgewick,1985).

ACKNOWLEDGEMENT

This project has been supported in part by a Faculty Development Microcomputer Grant through Drexel University.

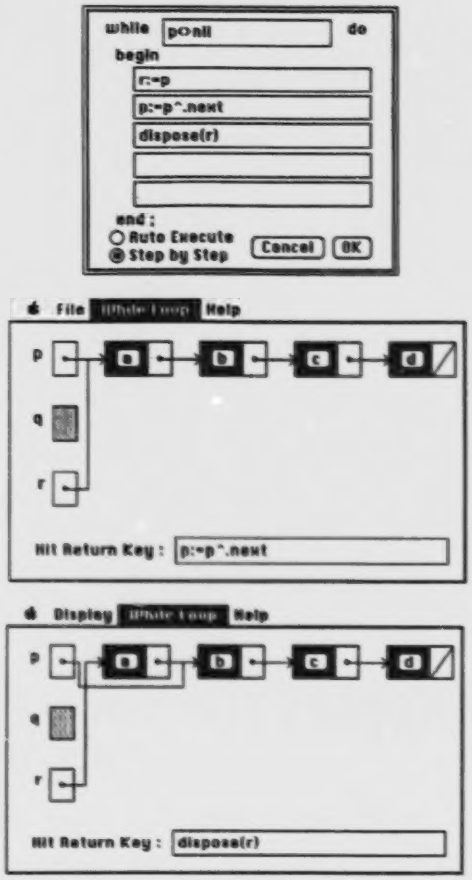
REFERENCES

Baecker,R., *Sorting Out Sorting*, 16mm. color sound film, 25 min., 1981.

Brown, M.H., and Sedgewick,R., "Techniques for Algorithm Animation," *IEEE Software*, pp. 28-39, January 1985.

Popyack,J.L., "Pascal Pointers and Dynamic Variables Demonstration Package," Kinko's Academic Courseware Exchange, 1986.

EXAMPLE



```
while p<>nil do
  begin
    r:=p
    p:=p^.next
    dispose(r)
  end;
```

○ Auto Execute
● Step by Step

Hit Return Key : p:=p^.next

Hit Return Key : dispose(r)

Conditional repetition using the while loop template. The user chooses step-by-step execution.

POLICY AND PRACTICE AFFECTING THE INSTRUCTIONAL USES OF COMPUTERS IN ELEMENTARY SCHOOLS

Robert L. Blomeyer, Jr. George W. Bright

Department of Curriculum and Instruction
University of Houston
Houston, Texas 77004

Abstract

This study looks at instructional policy and teaching practice influencing the educational uses of microcomputers in elementary schools. The research was a preliminary field study on the level of microcomputer use in the elementary schools of a single school district. A "computer literacy" program had been implemented two years earlier and the district's staff wanted feedback for program planning and devlopment.

The goals of the study were to identify (a) district and building-level policies impacting the instructional use of microcomputers, (b) the range of instructional activities involving microcomputers, and (c) the perceptions of staff concerning both future directions for microcomputers in the educational program and the support needed to make that future a reality. Although the effectiveness of the elementary instructional computing program was a question important to the district's administration, the researchers felt that gathering detailed information on the three goals listed above would enable a timely process evaluation by the appropriate district personnel. It was assumed that the availability of objective information on the program would lead to revisions in instructional policy and practice.

Background

Salomon and Gardner (1986) draw an important parallel between research on the instructional uses of television and research on the instructional uses of computers. In the research on instructional television, many investigators pursued questions about the "effectiveness" of TV as a delivery medium in comparison with other alternatives. The overall lack of significant findings from this earlier research may not be because TV is ineffective as a medium. Rather, it may be due to the lack of sensitivity of experimental studies to concomitant changes in the schools and classrooms where TV was used. The central point is that research questions about relative "effectiveness" (measured by comparisons of gains through testing) may be inappropriate. Salomon and Gardner recommend that research on instructional computer use might be more productive if it were organized using a heuristic approach to describe the context of computer use and to provide information useful for discovering specific questions that can be tested as hypotheses in subsequent research.

There are a few quasi-ethnographic studies that provide valuable precedents for conducting contextually relevant research on instructional computing. In particular, this research provides a core of "foreshadowed issues" that have great value for understanding our present situation. Smith and Pohland (1974) studied the effects of a community's dominant cultural patterns on the implementation of instructional computing. They found that the effective use of computer assisted instruction (CAI) in schools was not as simple as assigning the students time on computers to complete instructional tasks. Instead, CAI was found to be a complex mixture of "...physical, social, emotional and cognitive behaviors inextricably intertwined and shaped by both administrative routines and norms of the particular schools and classrooms" (p. 36).

In other pioneering research, Stake (1977) concluded that there are particular attributes associated with the role of the teacher in classrooms where computers are present that have a significant impact on whether students succeed or fail in adapting to the use of that new instructional medium. The students were fourth graders who studied mathematics on the CERL PLATO System (the University of Illinois at Urbana-Champaign). Students who lacked a large variety of study skills and learning strategies encountered severe difficulty and tended to give up without the intervention of the teacher. The teachers studied had apparently become well versed in the learning strategies necessary for the students' effective use of computers in the classroom. The study suggests that this probably was a result of immediate contact between the teachers studied and the developers and designers involved in the PLATO implementation project. This suggests that productive attitudes and management protocols for teachers can be learned through direct contact with educators who have become experienced with integrating computers into school settings. It is not clear, however, how well teachers can or will learn these things on their own.

More recently, case studies and other descriptive research have focused on staff training, hardware and software distribution, school conditions affecting access to computers, software design issues, and a broad concern for appropriate integration of instructional computing with prior curriculum content and objectives. For example, Sheingold, Kane, and Endreweit (1983) compared instructional computing programs in three contrasting school districts and Meister (1984) provided a single site case study of a "model" elementary school microcomputer program in the San Francisco area.

A multiple site case study conducted in southern California in 1983 (Boruta, M., Carpenter, C.R., Harvey, M., Keyser, T., LaBonte, J., Mehan, H., & Rodriguez, D.) suggests that there is reason to believe the use of microcomputers in school settings may contribute to either equalization or increased stratification of social groups within the population. This apparently depends on relationships between sources of funding for purchase of computers, categorical status of students using the computers (Chapter I, gifted and talented, racial minorities, etc.), the general type of instructional computer use (regular vs. special programs), and the general rationale for computer use in the individual district. Other multiple site case studies conducted over the last five or six years provide portrayals of a variety of instructional computing programs in individual schools and school districts (O.T.A, 1982; Blomeyer, 1985; Cline, H.F., Bennett, R.E., Kershaw, R.C., Schneiderman, M.B., Stecher, B., & Wilson, S., 1986; and Bennett, 1987). Related studies suggest that increased instructional flexibility attributable to individualization through the systematic use of computer assisted instruction is a significant attribute of some instructional computing programs (Blomeyer, 1986; McInerney, 1986).

District History

In our preliminary study, the twelve elementary schools in the cooperating district had students enrolled in grades K-5, with class size of about 25 students. About three years before this study, the school district had initiated the implementation of a "Technology Plan" to achieve the integration of computer technology into all the district's schools. The plan called for the initial development of instructional computing programs in the secondary schools. In school year 1982-1983 the cooperating district began to implement a secondary school computing program with the purchase of networked minicomputers to serve the programming, word processing, and vocational computing needs. In 1983-1984 the state announced a mandate for a middle school "Computer Literacy" curriculum and the district's implementation plan was modified to include installation of microcomputer laboratories in the middle schools.

In addition to existing programs in the secondary and middle schools, district personnel supported the idea that elementary students should also have an immediate introduction to computer literacy to maximize the effectiveness of the state mandated computer literacy classes. The state's curriculum guides for the middle school classes defined literacy as terms and concepts related to the history, social role, and use of computers; some indicated use of applications software; and computer programming in BASIC or Logo. When the mandated middle school programs were initiated, the state lacked clear guidelines for computing programs in the elementary schools. The general direction of the middle school mandate suggested a high probability of similar state regulations requiring "computer literacy" classes in the elementary schools. Subsequently, the cooperating district's elementary computer literacy curriculum focused primarily on a terms-and-concepts approach augmented by some hands-on experience as per the perceived state direction.

The district purchased two portable microcomputer laboratories that were to be used to implement the computer literacy program. According to district personnel, these initial portable labs were to be an "initial stimulus" to the growth of more comprehensive microcomputer programs in the elementary schools. Because no appropriate models or clear state guidelines were apparent, the district took a "low profile" approach which was intended to encourage incremental development of instructional computing in the elementary schools.

The elementary microcomputer program was initiated in 1984-85 with fourth and fifth grade students and was to be expanded in 1985-86 with the addition of third grades in each elementary school. The mobile labs consisted of networked microcomputers with a limited amount of CAI (generally drill and practice exercises), graphics, games, and Logo software. Initial training had been offered to teachers throughout the district when the mobile microcomputer labs were first purchased. This training consisted of brief introductory experiences using the microcomputers, experience with a sampling of the software that was to be provided along with the microcomputers, and an introduction to the substance of the computer literacy classes.

During 1986-86 when this study was conducted, fourth and fifth grade teachers were teaching curriculum units on computer terms and concepts for a limited period of time in each elementary school. The particular period of time when each class used the mobile labs depended on the rotating schedule which moved the labs between schools. An exit exam was developed for use in all elementary computer literacy classes. At the time of the study, only teachers in the fifth grade classes were reportedly using the exit exams. The substance of the exam was generally recall and recognition of computer terminology. The elementary computing curriculum was intended by the school district to be an addition or an "enrichment" to the existing elementary school curriculum.

Procedures

To find out about the instructional use of microcomputers in the district's elementary schools, four of the twelve schools were chosen for site visits including limited direct observation and interviews with school personnel. In addition, a survey on school microcomputer use was developed and mailed to all twelve elementary schools in the district. Interviews and observation were planned as the primary data source for the study with the survey establishing an ethnographic baseline to supply context illustrating the generalizability of the particular observations to more general conditions that might be affecting elementary schools in the entire school district. The surveys were administered to three individual staff members in each elementary school: principal, the school's designated microcomputer resource person, and one non-computer using teacher. Of the 36 survey forms that were mailed out, 31 were completed and returned.

The schools chosen for the site visits were selected to cover the widest possible range of scenarios, including both variations in perceived levels of local school receptivity to the microcomputer program and variations in the affluence and ethnicity of the school district. Typically, each initial visit began with a 30-60 minute interview with the school's principal. In these, the investigators typically got a general sense for the characteristics of the computer activities in the schools and identified teachers or other personnel who were key players in each school's evolving microcomputer program. After the initial interviews with the principals, the investigators attempted to schedule meetings with all the relevant staff members. These interviews progressively refined the focus on events and attitudes that characterized the particular "set" toward microcomputer use in each school.

In total, eight one-half day visits were made for interviews and limited direct observation in four elementary schools. In addition, two half-day visits were made for interviews with district curriculum consultants concerned with instructional programs in mathematics, language arts, reading, and other curriculum areas.

After completion of all observation and interviews, a preliminary analysis of the data was undertaken by the cooperating researchers. Observed events and interview data illustrating forshadowed questions regarding policies, the range of computer activities, and the nature of staff attitudes toward the program were categorized for comparison with the information obtained from the survey. Interviews with administrative staff and classroom teachers, the survey data, and field notes from the limited amount of direct observation showed general agreement on a range of issues specific to policy and practice in the elementary microcomputer program.

Results: Observation and Interviews

In all four schools visited, the fourth and fifth grade teachers professed to be somewhat disillusioned by the current state of the computer literacy classes. Because of staff turnover, many of the current teachers had never been given the initial inservice training on either the hardware or the elementary technology curriculum. Even those who had been trained, frequently commented that their introduction had not been sufficient and more training was needed. Generally, the emphasis on terms and concepts was questioned because of its limited scope. Many teachers interviewed also complained about the unreliability of the hardware and the generally low quality of the instructional software available with the traveling microcomputer laboratories. Some teachers voiced open opposition to the computer literacy curriculum and said that they would rather have that time to work on subject matter required by the state scope and sequence. A common remark was that the technology curriculum was seen as an "add on" that was not specifically relevant to the state mandated scope and sequence for elementary schools.

Some teachers indicated that they made attempts to use the drill and practice software that accompanied the traveling labs as a supplement to regular classroom instruction. Those teachers reported unsatisfactory results from these attempts at curricular integration because of the inflexibility of the software and the infrequent availability of the labs in the schools.

Teacher accounts as well as observations classes in schools where Apple II microcomputers had been purchased by local PTA funds, indicated more integration of these microcomputers in curriculum. This was especially true for the one school studied where Apple IIe microcomputers had been supplied as original equipment for a designated microcomputer laboratory. Use of the Apple II microcomputers in support of general curriculum objectives was generally limited, however, to drill and practice in mathematics and spelling.

There was some observed use of word processing and data base management activities with the gifted and talented classes, but the use of these tool programs was marginal compared to the use of more traditional CAI. Teachers working as resource specialists with students having learning disabilities and other handicaps had microcomputers and some instructional software available for their use. These teachers indicated that word processing software would be a welcome addition to their programs, but they would need more training to use it successfully with the students.

Although many of the teachers expressed an interest in the use of flexible tool applications, they apparently lacked training and easy access to hardware and software applications. Both the curriculum specialists and the classroom teachers regularly expressed the opinion that if the district wanted to develop a stronger microcomputer program in the elementary schools they should purchase Apple II microcomputers and a variety of instructional applications for regular use in the elementary schools.

In the four schools visited during the study, patterns of observed microcomputer use varied markedly. Probable factors influencing individual school microcomputer programs were number of available microcomputers, hardware installation (fixed or mobile), availability of software, scheduling efforts by the principal or a designated microcomputer resource person, availability of parent volunteers to assist with microcomputer use, and overall attitude of the local community toward the instructional use of microcomputers in each school.

One school had a central microcomputer laboratory (15 Apple IIe microcomputers) where each student used mathematics and spelling drill and practice software for 30 minutes per week. Gifted and talented classes in that school were pilot testing The Voyage of the Mimi which includes videotaped story episodes, workbooks, teachers guides, and microcomputer activities suitable for both social sciences and general sciences curriculum.

In another school there were seven microcomputers; one was assigned to the Learning Resource Center and remaining units were on rolling carts and assigned to individual teachers in grades K-5 on a rotating schedule. Some teachers used the microcomputers for drill activities but others let the microcomputers sit idle.

In a third school, three Apple microcomputers were housed in the learning center. These were primarily used as "rewards" with students on behavior modification management plans. In addition one grade two teacher used drill and practice software for about 15 minutes per child every other week.

In a fourth school, only two microcomputers were available for instructional use. One was used primarily for administrative tasks (e.g., grade books, P.E. record keeping) and the other was available only for use with Title I students. This scarcity of resources created a climate where access was limited, demand was high, and tempers were short.

Teachers interviewed in the four schools studied seemed to support the idea of instructional computing, but there was some apparent confusion about the district's policy on the role of microcomputers in the elementary schools. Gifted programs and remedial education were viewed as more important uses of microcomputers than computer literacy. Though the teachers reported that they favored the integration of microcomputers as a supplemental medium for instruction in a variety of curricular areas, the district policy was oriented to a terms-and-concepts approach. This conflict between the teachers' commitment to the traditional scope and sequence and the apparent "non-centrality" of the district's approach to microcomputers in the curriculum had apparently been unrecognized or was not viewed by the school administration as a significant issue affecting the elementary instructional computing program.

Other significant issues were the teachers' perceptions that additional inservice training was necessary to develop the appropriate classroom use of microcomputers, uneven access to microcomputer resources within and among the schools (i.e., fourth and fifth grade students having more systematic access to microcomputers and schools in more affluent suburbs having more hardware units because of PTA donations), and the apparent restriction of microcomputer applications promoting higher level thinking and problem solving skills (e.g., creative uses of tool programs and programming languages) to the small population of gifted students.

It appeared that the instructional staff (teachers and principals) was at odds with the district's existing position guiding the introduction of microcomputers into the elementary schools. Although the computer literacy initiative spurred the acquisition of hardware and software by the individual schools, the extension of microcomputer use into the regular scope and sequence of the curriculum was slow at best. The teachers seemed to have been looking for cues from the curriculum materials provided with the traveling labs to help them determine what they should be doing with microcomputers in the curriculum. The cues actually received apparently encouraged teachers to try limited use of drill and practice CAI programs and to provide instruction about computer terminology and concepts. An apparent lack of inservice training or other educational incentives has left the teaching staff with little base on which to build a more innovative instructional computing program.

Results: Survey

Survey data generally seemed to support the anecdotal observations made in the individual schools. Generally, 5-15 microcomputers were permanently assigned to each school in the district. These were located in central sites, individual classrooms, or on rolling carts. The most important use of computers was perceived to be middle school computer literacy, programming or computer mathematics at the secondary school level, and the elementary technology curriculum. However, some staff members indicated that they did not believe the district had a clear policy guiding the use of microcomputers in elementary curriculum.

Respondents in the elementary schools also indicated that the temporary assignment of movable labs was not satisfactory. Regular access to microcomputer equipment that would be permanently assigned to the schools was seen as necessary. Purchase of instructional software appropriate for regular use with the elementary classes was also seen as a necessary addition. Respondents who identified themselves as elementary teachers indicated that the microcomputer training conducted by the district at the outset of the technology program was inadequate and additional training would be necessary to improve on the present situation.

District Actions Initiated After Completion of the Study

The field study detailed above was completed by early May 1986, and a final report was delivered to the district in July 1986. A reply from the district in December 1986, indicated that the rotating microcomputer labs were retired and that over the summer enough Apple IIe's were purchased to "bring everybody up to the same level." All twelve elementary schools in the district are now reported to have a minimum of twelve Apple IIe's to be used for CAI.

In addition, $3000 was allotted to each elementary school for the purchase of software. A prerequisite for release of these funds to a school was the development of an implementation plan demonstrating how the microcomputers and software would be used. The instructional personnel in each school are responsible for evaluating the software prior to purchase. Copies of evaluations are forwarded to the school's central administration so that a list of approved and disapproved titles is compiled each month and circulated among the schools to eliminate duplication of effort. Any disagreements over software evaluations are rationalized by the district's Curriculum Department.

The district made the decision to increase support to the instructional computing program in the elementary schools during the spring of 1986. Members of the school district staff said they were aware that there was an inequitable distribution of microcomputer resources within the elementary schools because of uneven purchases of microcomputers by groups supporting individual schools. Microcomputers purchased by PTA's activity funds or other sources were allocated within particular schools to fill the needs of categorical programs (gifted and talented, special education, Chapter I, etc.). In all, about $250,000 was spent over the summer of 1986 to equalize the uneven distribution of microcomputer resources and to make instructional computing resources available to support subject matter instruction in the district's elementary schools.

Conclusions

While only tentative conclusions are warranted by this preliminary study, the evidence suggests that states and individual school districts should be careful in planning and structuring the initial policies and interventions used to introduce instructional computing into the elementary curriculum. In this case, the state's lack of clear policy guidance for elementary instructional computing programs led to the school district's adoption of a "terms and concepts" approach to elementary school computer classes. This initial program had an apparently negative affective impact on elementary school teachers and administrators. An important finding of the study is that in this specific case there was a clear conflict between the instructional goals of the classroom teachers (subject matter instruction as specified by the state's elementary scope and sequence) and the district's more limited goals for the elementary computing program. This suggests that state education agencies and individual school districts should make the curricular integration of instructional computing, with a concentration on content emphasized in the general curriculum, a high priority among program objectives taken into account when planning for the use of microcomputers in elementary schools.

In the four schools studied, the parents in the community were evidently "pro-technology" and supported the introduction of microcomputers in elementary curriculum. In some schools, parents wanted more microcomputers in the schools than the district could afford, so the parents purchased Apple II microcomputers with PTA funds. Because of this, schools in more affluent areas generally had more microcomputers than those in less affluent areas of the district. Parental intervention at the local school level led to uneven distribution of resources and possible programmatic inequity.

The characteristics of initial interventions and the perceived nature of a state's or district's policy guiding the integration of microcomputers in elementary curriculum is likely to profoundly affect the range of options that teachers see for microcomputers in the classroom. A definition of computer literacy as knowledge about terms and concepts was seen by instructional personnel and some principals as being irrelevant to the primary educational mission of the elementary schools. Additionally, teachers' perceptions of computer literacy programs as being peripheral to existing curriculum objectives may be difficult to alter once such programs are established. Unless general policies supporting the integration of microcomputers as a relevant and flexible instructional medium are established at the outset, implementation of comprehensive microcomputer education programs in elementary schools may be difficult.

References

Bennett, R.E. (1987). Planning and evaluating instructional computing programs. Columbus, Ohio: Merrill Publishing Company.

Blomeyer, R.L. (1986). Instructional policy and the development of instructional computing: Maintaining adaptive educational programs. Educational Considerations, 13(3), 17-20.

Blomeyer, R. L. (1985). The use of computer-based instruction in foreign language teaching: An ethnographically -oriented study (Doctoral Dissertation, University of Illinois, 1985). Dissertation Abstracts International, 46, (07A), 1909.

Boruta, M., Carpenter, C.R., Harvey, M. Keyser, T., LaBonte, J., Mehan, H., & Rodriguez, D. (1983, July). Computers in schools: Stratifier or equalizer ?. In J.A. Levin & R. Souviney (Eds.), The Quarterly Newsletter of the Laboratory of Comparative Human Cognition, 5(3), 51-55. San Diego: University of California Laboratory for Human Information Processing.

Cline, H.F., Bennett, R.E., Kershaw, R.C., Schneiderman, M.B., Stecher, B., & Wilson, S. (1986). The electronic schoolhouse. Hillsdale, N.J.: Lawrence Earlbaum Associates.

McInerney, W.D. (1986). Educational computing: Some policy implications for administrators. Educational Considerations, 13(3), 21-25.

Meister, G. (1984). Successful integration of microcomputers in an elementary school (Project Report 84-A18, Institute for School Finance and Governance). Stanford CA: Stanford University. (ERIC Document Reproduction Service No. ED 256 059)

Office of Technology Assessment (1982, November). Informational technology and its impact on American education. Washington DC: U.S. Government Printing Office.

Sheingold, K., Kane, J., & Endreweit, M. (1983). Microcomputer use in the schools: Developing a research agenda. Harvard Educational Review, 54, 412-432.

Smith, L. & Pohland, P. (1974). Educational technology and the rural highlands. In R. Stake (Ed.), Four evaluation examples: Anthropological, economic, narrative and portrayal (AERA Monograph Series on Curriculum Evaluation No. 7, pp. 5-54) Chicago: Rand McNallay.

Salomon G. & Gardner, H. (1986). The computer as educator: Lessons from television research. Educational Researcher, 15(1), 13-19.

Stake, B. E. (1977). PLATO and fourth grade mathematics. Urbana IL: Center for Instructional Research and Curriculum Evaluation, University of Illinois. (ERIC Document Reproduction Service No. ED 228 040)

THE ROLE OF SCHOOL-BASED COMPUTER COORDINATORS AS CHANGE AGENTS AND STAFF DEVELOPERS

Neal Strudler

College of Education
University of Oregon
Eugene, OR 97403

ABSTRACT

School districts are committing large amounts of money to teacher training, equipment, and instructional materials in the hope that implementation of instructional computing will occur. To facilitate the implementation of this innovation, some schools have created a new role: the school-based computer coordinator. This research study seeks to derive a clearer understanding of that role by conceptualizing such coordinators as change agents and analyzing their strategies, skills, and outcomes using a framework developed by Matthew Miles.

Questionnaires and in-depth semi-structured interviews were used to gather data from the computer coordinators, their supervisors, teachers, and parents. Interviews were tape-recorded and selected segments were transcribed. Data were analyzed case by case, and then across cases.

The present findings suggest that school-based computer coordinators perform a variety of functions that facilitate computer use. While prospective coordinators should be well-versed in their knowledge of instructional computing, it appears critical that they possess strong interpersonal and organizational skills to carry out necessary training and support functions.

The findings and conclusions of the study are consistent with previous research on the characteristics of effective change agents and staff developers. Because case study methodology was used, however, the findings should be replicated using larger samples and more objective methodology, especially in the measurement of implementation effects.

INTRODUCTION

School districts are committing large amounts of money to teacher training, equipment, and instructional materials in the hope that computers will be used as an effective tool throughout the elementary school curriculum. To facilitate the implementation of this innovation, some schools have created a new role: the school-based computer coordinator. Little is known, however, regarding the specifics of that role. How do computer coordinators work with teachers? What strategies do they employ? What skills do they need to be effective?

The purpose of this study is to examine the role of school-based computer coordinators as change agents in elementary school programs. The study replicates aspects of the research project Patterns of Successful Assistance in Urban School Improvement Programs (Miles, Saxl, & Liberman, 1985) conducted at the Center for Policy Research and Teachers College, Columbia University.

Background for the Study: The Change Agent Role

A recent study of school improvement found that the successful implementation of innovative programs is dependent upon the intervention of key personnel involved in various change agent roles (Huberman & Crandall, 1982). A change agent is defined as an assistance person whose role includes responsibility for facilitating change. This emerging role is being given increased attention because of a shift in the approach to school improvement. Whereas the post-Sputnik school reform campaign tended to rely on centralized curriculum changes, current efforts focus on school-based change. With this change there has been a corresponding shift in change agent roles from a disseminator of ideas and materials to a process consultant (Beaton, 1985).

While change agents are widely used as "special assisters" in current school improvement programs, little is known about their functions, the strategies they employ, the specific skills most crucial for success, or how to train for such skills (Fullan, 1981; Miles, 1984; Miles et. al., 1985). The Patterns of Successful

Assistance study (Miles et. al., 1985) focused on these concerns in its two-year investigation of 17 change agents working in three urban school improvement programs. A synthesis of findings resulted in 18 key skills for educational change agents.

Beaton (1985) replicated aspects of the Patterns of Successful Assistance study in focusing on the role of staff development specialists in three school improvement programs. He found that these specialists use a combination of product- and client-centered strategies that require sophisticated interpersonal skills. The main teacher outcomes he reported were motivation, positive attitudes, and enhanced communication and cooperation between teachers.

The present study extends the findings of Miles and his colleagues and Beaton. It replicates aspects of their studies to investigate a new change agent role: the school-based computer coordinator.

METHOD

Design. This is an exploratory case study that uses questionnaire, interview, and observation procedures to survey three change agents carrying out a school-level computer coordinator role.

Sample. Eugene School District 4J currently has nine elementary schools with released-time computer coordinators. Time allocated for this role ranges from .1 to .6 FTE. A sample of three of these schools was selected for this study because their coordinators had been identified as having brought about a high degree of implementation of instructional computing. The identification process involved consulting with the district computer coordinator, the district evaluation specialist, and the Educational Service District curriculum/staff development specialist, and by personal observations of computer implementation in the schools. Additionally, consideration was given to choosing coordinators who vary with regard to critical variables, namely: (1) technical expertise in computing, (2) role expectations of computer coordinators by supervisors, teachers, and themselves, (3) previous teaching experience in the particular school, and (4) available resources.

Data Collection Instruments. Semi-structured interviews and a questionnaire were adapted from those used in the studies of school improvement coordinators by Miles and colleagues (1985) and of staff development specialists by Beaton (1985). A form of the questionnaire was given to each coordinator to assess his or her strategies and skills, and achieved outcomes. Another questionnaire, almost identical, asked each supervisor to prepare a similar profile of each coordinator. Four interview schedules pertaining to the role and qualities of the computer coordinators were administered. Informants included the coordinators (interviewed twice), their supervisors (principals, district adminstrators), and their clients (teachers and parents). Fifty-two interviews were conducted, ranging from twenty-five minutes to more than two hours in length. Further data were gathered by direct observation of instruction and staff meetings and by analysis of planning documents.

The data were collected over a ten-week period of time.

Data Analysis. All 52 interviews were tape-recorded, and selected segments were transcribed. Field notes and interview data were coded for analysis according to the list of variables developed by Miles. Data were analyzed case by case, and then across cases. Finally, the results will be compared with those of Miles and his colleagues (1985) and Beaton (1985).

RESULTS

The data analyses completed thus far have yielded the following findings:

1. The role of the computer coordinator is multi-faceted and difficult. Key functions that coordinators fulfill include:
--securing hardware and software through allocated funds, grant writing, and fund raising activities.
--ordering, organizing, and managing all computer resources
--modeling effective teaching.
--training teachers and conducting workshops.
--serving as a resource person to teachers for choosing software and planning instructional computer use.
--recruiting and training parent volunteers.
--facilitating school-wide planning in conjunction with building administrators and the school's computer committee.

2. Effective coordinators display a variety of styles, skills, and strategies. All informants, though, seem to prefer a facilitative style, characterized by a collaborative, supportive, and collegial approach. This style appears to be especially effective when complemented by good administrative and organizational skills. Though it involves a difficult balance, effective coordinators seem to combine being patient and "low key" with being active and systematic.

3. Nearly all informants expressed that interpersonal skills for coordinators are more important than their knowledge of computing. Supervisors added that "people" skills are more difficult to teach and learn than are computing skills. Thus, they feel that people chosen to be coordinators should already

possess strong interpersonal skills.

4. Effective coordinators take into account the "plight" of the classroom teacher. Elementary teachers are overloaded with the demands of a crowded curriculum and resist further demands. Coordinators help by: (1) screening for appropriate software, (2) having computer labs "pre-booted" and ready for student use, and (3) assisting with custodial chores. Teachers seem much less resistant to using computers in the curriculum if instructional time with students is maximized.

5. Coordinators face a conflict between overcoming teacher resistance and facilitating teacher independence. Ambitious plans for getting teachers to independently "fly on their own" may result in increased teacher competence or greater teacher resistance. Effective coordinators consider the comfort level and commitment of the staff and administration in choosing a preferred time-line and approach for computer implementation.

6. It appears to be an advantage to have a coordinator who has been chosen from the existing staff of a school. With a sense of rapport and trust in place, that coordinator may be able to be more ambitious in moving the program along. A coordinator chosen from outside the staff has a heightened need for the interpersonal skills to effectively collaborate with a new faculty. If that staff is committed to expanded computer implementation, however, a skilled coordinator appears to experience few, if any, obstacles in adjusting to the new school situation.

7. The role of computer coordinators will likely change with the level of computer use and teacher expertise in a particular school. When teacher comfort with computers is low and resistance is high, a coordinator must "win over" teachers with effective "modeling" of classroom instruction and an encouraging but patient style. A coordinator's teaching skills and "bedside manner" may become less critical as teachers become more comfortable with educational computing.

8. A .5 FTE seems adequate for the computer coordinator of a medium-to-large sized elementary school. If less time is allocated, role expectations should be adjusted accordingly.

9. Outcomes realized due to the work of released-time computer coordinators include: the improved skills of teachers and the readiness for further advances, the implementation of program goals, teacher satisfaction with the program, and teachers' increased feelings of self-esteem and professional growth.

Data analyses in progress should yield additional findings on the strategies and skills used by this particular group of change agents to bring about implementation of computers in schools.

DISCUSSION

The present findings suggest that school-based computer coordinators perform a variety of functions that facilitate computer use. While prospective coordinators should be well-versed in their knowledge of instructional computing, it appears critical that they possess strong interpersonal and organizational skills to carry out necessary training and support functions.

The findings and conclusions of the study are consistent with previous research on the characteristics of effective change agents and staff developers. Because case study methodology was used, however, the findings should be replicated using larger samples and more objective methodology, especially in the measurement of implementation effects.

SIGNIFICANCE OF STUDY

This research project has provided increased understanding of change processes in education by identifying the characteristics of personnel who are effective in these processes. The characteristics identified in this project can be compared with characteristics identified in other studies to determine whether there are generally effective characteristics or whether characteristics are specific to a particular type of change.

The project's findings also have practical significance. Information on successful strategies used by computer coordinators might increase their effectiveness as change agents and enable their supervisors to provide better guidance and support. Furthermore, identification of the skills required in that role could aid in the selection and training of prospective coordinators.

REFERENCES

Beaton, C. (1985). Identifying change agent strategies, skills, and outcomes: the case of district-based staff development specialists. Unpublished doctoral dissertation, University of Oregon.

Fullan, M. (1981). School district and school personnel in knowledge utilization. In R. Lehming & M. Kane (eds.), Improving Schools: Using What We Know (pp. 212-252). Beverly Hills: Sage.

Huberman, M. & Crandall, D. People, policies, and practices: Examining the chain of school improvement, Vol. IX, Andover, Mass.: The Network, Inc.; 1982.

Miles, M. B. (1984, April). The role of the change agent in the school improvement process.

Abstract of paper presented at the annual meeting of the American Educational Research Association, New Orleans.

Miles, M. B., Saxl, E. R., & Lieberman, A. (1985). *Key skills of educational "change agents": an empirical view*. Manuscript submitted for publication.

THE DREXEL UNIVERSITY - PRISM COMPUTER APPLICATIONS IN TEACHING PROGRAM: A STUDY IN UNIVERSITY, SCHOOL DISTRICT, AND INDUSTRY COLLABORATION

Francis M. Betts III, Ed.D., Executive Director, PRISM
P.O. Box 58698, Philadelphia, PA 19102
Elizabeth L. Haslam, M.Ed., Program Manager, PRISM Documentation Project
Fredricka Reisman, Ph.D., Director of Teacher Education
Drexel University, Philadelphia, PA 19104

Abstract

The Computer Applications in Teaching Program (CAITP) is a 24-month, 3-phase program intended to increase the effectiveness of instruction in secondary science and mathematics in the Philadelphia Public Schools. When completed in the summer of 1987, the program will:

1. Have trained more than 200 high school teachers in the use of the microcomputer and the development of computer applications in science and mathematics instruction;
2. Developed more than 20 exemplary applications related to the curriculum being taught in the schools;
3. Linked 38 schools into an electronic network for resource sharing;
4. Benefited more than 12,000 students;
5. Established a national model of industry-education collaboration for staff development in the introduction of computer technology into science and mathematics instruction through teacher-developed applications. The total cost of the program, approximately $200,000, was raised from foundations, public and private sources other than the School District operating budget. This case study reports on the progress and outcomes based on analysis of the documentation of quantitative and qualitative indicators.

Introduction

The application of computer technology to instruction in science and mathematics is generally regarded as a high priority goal of education. The importance of technology as an instructional resource has been emphasized in A Nation at Risk (1983), as well as other studies advocating redesign of the science and mathematics curriculum (National Commission on Excellence in Education, 1983; Boyer, 1983; National Science Board Commission on Pre-College Education in Mathematics Science and Technology, 1983).

In 1985 the Commonwealth of Pennsylvania mandated a program of computer literacy that emphasized acquisition of facility in programming. At the local level this mandate resulted in the concentration of available hardware into computer labs. District staff training plans concentrated on developing skills almost exclusively used for instruction in computer programming. During the same time period, increasing concern over the potential for liability litigation increased the need for the use of computers in the science classroom for simulations to demonstrate potentially dangerous or lengthy experimental events. Thus, at a time when the need for developing computer applications in science and mathematics instruction was increasing, the availability of hardware in the Philadelphia public school classrooms and the resources to develop teacher skills in the uses of technology in instruction was decreasing.

In order to address this issue, a three-phase program to more fully integrate computing into the instructional process in the School District of Philadelphia was undertaken as a collaborative effort betweeen the School District, Drexel University, and the Philadelphia Renaissance in Science and Mathematics (PRISM), representing the universities, corporations, and non-profit cultural institutions of the region.

The goal of the program is to increase the effectiveness of instruction in science and mathematics through more effective use of technology in instruction. This goal is being attained by:

1. Introducing secondary mathematics and science teachers to the uses of the microcomputer as an instructional tool in a "hands-on" environment;
2. Encouraging the development of applications of microcomputer technology in instruction; and,
3. Establishing a continuing means of disseminating information and sharing applications via telecommunications linkages between schools.

The anticipated outcomes from this program include:

1. Improvement of the instructional process through the increased use of microcomputers and software specifically related to the School District curriculum in science and mathematics;
2. A more knowledgeable and technically oriented group of department heads and teachers who can provide leadership within their school to help integrate the use of computers into instruction.

The Computer Applications in Teaching Program (CAITP) was organized in three stages in order to be more sensitive to the phenomenon of change by developing expertise and leadership from within the teacher population, rather than imposing "expert opinion" (Sarson, 1971).

Phase 1

Phase 1 was conducted by Drexel University under the direction of Dr. Fredricka Reisman in the Fall of 1985. It consisted of 15 weekly, two-hour sessions for 38 mathematics and science department heads who enrolled voluntarily and without pay. Sessions were held at two high school sites provided by the School District at which each participant had the use of an Apple IIE microcomputer.

The first eight sessions emphasized historical development, instructional design concepts, hardware configuration and software selection. An introduction to Appleworks was also begun, since this was the kind of applications' software provided to all schools.

A midway evaluation indicated a need for more "hands-on" activities and a closer integration of the material with perceived needs. Consequently, the remaining sessions were directed toward lecture-demonstrations of software related to a specific instructional objective in the curriculum or to a teacher-identified need. Dr. Francis M. Betts III, Executive Director of PRISM, was asked to join the management and instructional team. Dr. Betts had worked previously in industry as a training consultant and systems integrator.

At Dr. Betts' recommendation, the training design was modified to include a comparative analysis of the solution of instructional problems using commercial software contrasted to Appleworks' solutions developed by Dr. Betts and distributed to the trainees as templates. The applications are shown in Figure 1.

Each week an attempt was made to demonstrate a commercially developed application in at least two of the three subject areas and a similar application based on the Appleworks' package. In addition to a lecture-demonstration component, the department heads received Appleworks' templates on diskettes. Applications presented included computer interfacing to demonstrate endothermic and exothermic reactions; the use of scientific data bases; graphic solutions of simultaneous equations; classroom management applications, such as test scoring and grade reporting; and a special session on telecommunications. At least one hour of each session was devoted to "hands-on" activities.

The chief problem in Phase 1 was the difference in skill levels of the participants. Those department heads who were already active computer users quickly became frustrated and attendance within this subgroup dropped significantly in the early weeks. At the other extreme were department heads who had little or no experience, e.g., they had never seen the inside of the computer before, and most had never "booted" a computer from a cold start. Summative evaluation by the participants who completed 5 or more sessions indicated significantly more positive attitudes toward computing than those who attended infrequently.

Phase 2

The goal in Phase 2 was to extend the instructional benefits to teachers in the schools and to initiate development of exemplary applications for subsequent dissemination throughout the system. Four main activities were accomplished in Phase 2:

1. PRISM established an electronic bulletin board and mail system, "High Tech Talk," and began distribution of modems to mathematics and science departments. This activity will continue during Phase Three. All secondary math and science departments will have received telecommunications hardware and software by May 1987.

2. PRISM development grants of up to $5000 per school were given to nine schools, based on competitive proposal submission. Eligibility to submit a proposal was based on Phase One participation. Proposals were reviewed by School District and university personnel.

3. Drexel University organized and conducted an intensive 2-week summer institute for 59 teachers from the nine schools which received the PRISM grants. The School District loaned each teacher a computer for use at home during the summer. PRISM equipped these computers with modems and an active sharing of information continued throughout the summer after the formal sessions were completed.

4. In order to continue the momentum generated during the summer, Phase Two continued in the Fall with two mini-conferences at which the teacher-teams from each school presented the results of their activities. These conferences, held on September 23 & 25, 1986, were a sharing with the other institute participants of preliminary outcomes and concerns. On November 8, 1986, the teacher-teams made public presentations on their "work in progress." Completion of the PRISM grant-supported projects in the Spring of 1987 is being facilitated by continuing on-site consultation provided by Dr. Fredricka Reisman of Drexel University and Elizabeth Haslam, Documentation Manager of PRISM.

Phase 2 Evaluation

The Phase 2 evaluation, carried out by Elizabeth Haslam, indicated some very beneficial preliminary outcomes. Cohen's (1970) study indicates that growing perceptions of competence among teachers have an influence on their microcomputer implementation proneness in a positive direction, as well as increasing their interest in, and enthusiasm for improving instruction through the use of computers. An evaluation design combining qualitative and quantitative methods was used to assess the two main project goals. The methods included:

1. Participant observation - the evaluator as participant observed the institute's activities daily and conducted informal interviews. Notes were recorded in a daily log and later analyzed.

2. A systematic differential instrument was developed for teachers to assess the specific objectives dealing with using and evaluating software.

3. Beall & Harty's (1979) survey instrument was used for assessing attitudinal changes of participants.

The evaluation design was expanded to provide feedback to the program participants in order to enhance their perceptions of competence. Therefore, the evaluator also:

1. Assisted in individual school evaluation

aspects of the CAITP instructional design wherever such assistance was needed;

2. Provided materials for background reading on collaborative research methods to the trainees; and

3. Conducted site visits prior to the November 8th conference.

Discussion

Survey. There are clear indications from the participants' self-ratings in the proneness scale that the attitudes toward computer use changed. These results are shown in Figure 2. In items 2, 3 and 5, the mean scores showed an increase of one level or more.

Although the pretest and the posttest were separated by only 10 days of instruction, the institute's activities resulted in participants' developing increased positive attitudes toward using computers for instructional planning.

Electronic network. Participation in the electronic network was reviewed by examining the patterns of on-line activity.. These patterns are shown in Figure 3. In the overview of the individual school patterns of on-line activity, two patterns emerged. The first was that the amount of use was not related to the *size* of the school team; frequent users came from both small and large departments. Although Mastbaum and Roxborough (with three members each) had high activity on-line in minutes, and showed a characteristic cohesiveness in their projects, both Central and George Washington (averaging 8 members each) also reflected frequent on-line activity.

The participants who were active users of the network increased the communications increased communication within their departments. For example, Roxborough teachers, when interviewed, said that they used the network to inform each other about upcoming meetings, purchases of software, planned activities and what they were learning in the institute. The department head said, "Computer talk made us feel like a group; it was special way to communicate."

A second pattern was that, in several instances, a key person within a department who exhibited a high level of online activity, also became the *de facto* leader of the group in project development. In some cases this person was not the designated team leader, as evidenced at the September departmental presentations and site visits.

Although the modems were returned at the close of the summer institute, several teachers purchased modems at that time in order to use their new skills in on-line networking.

Discussion of the Observations

Planning Sessions.

The early planning sessions for department heads (June 23) of the schools receiving PRISM grants provided an overview for the entire Phase 2. At this meeting, although the agenda covered the institute and follow-up activities, the department heads expressed some concerns about whether their curriculum projects could be successfully developed and implemented. They had a "wait and see" attitude.

Most of the informal talk among department heads concerned "directives" on curriculum that they felt were an impediment in a project such as CAITP. One department head said that directives on curriculum were often contradictory. "I get so frustrated; we are never asked--we're *told*," she said.

Having completed Phase 1, however, most of the department chairs were impressed with the total picture, especially with the continuity of activities and field support planned for the entire project.

Institute In-Service Summer Training. It was evident that the 59 participants in this summer institute were very interested in the computer application projects and in receiving instruction. The trainees had a wide diversity in computer backgrounds, but all were anxious to get started. From the observations over the period of the institute and the informal interviews, several themes emerged:

1. *The focus of the participants changed from the logistics of getting to the institute in the first week to computer concerns in the last week.* In the first week problems with parking seemed paramount; even when Drexel University's Vice-President Gemmill welcomed the group, the questions to her were about tokens and available parking spaces. Other problems included how to set up the computers, how to gain access to software and distribute the modems. Some of the teachers seemed anxious about what they were supposed to know and confused about the agenda, even though each group had a specific project to develop.

Gradually, by the second week, the participants began to interact more with the instructors, staff and each other. In the last week, the participants were joking about the computer "experts" among them.

2. *The "hands-on" activities were preferred to the professional presentations.* During the nine days of presentations and "hands-on" activities, the workshop sessions which combined these two aspects were well received. The telecommunications' workshop was rated very favorably, but some of the presentations by commercial suppliers were not. Most complaints had to do with needing more "hands-on" activities earlier in the institute and fewer guest speakers.

3. *Project development varied considerably depending on each department's communication and level of interactions.* Some of the department heads and teachers had excellent rapport and communication; in those cases, the curriculum project was developed quickly. In other cases, valuable time was lost in trying to find a concrete focus and to apply the institute's instruction to the project goals.

Several teachers had expected instructions and manuals for *Appleworks*. That resource did not materialize because the School District had purchased 10-packs with only one manual per pack. As a result the teachers who had counted on this resource were disappointed.

There were questions about how other schools in the country were using computers to improve instruction. Later, several articles on this topic were distributed to the trainees. Almost all the handouts were immediately snapped up. The participants even asked for additional copies to take back to their home schools.

Other questions had to do with time constraints. When would they be able to work on planning their instructional projects? How could they implement their projects with limited access to hardware? How could the scheduling be facilitated? At several points during the institute, the departments met for brief discussions of how to integrate the instructional focus of "hands-on" activities with evaluation components. The institute generated many more questions of instructional design than it had time to resolve.

4. The most visible achievement was the telecommunications goal of having everyone log on; contrasted with the separate curricular goals, this goal was concrete and immediately attainable. This telecommunications goal was met successfully: on July 11, 59 modems were given out, and 42 participants signed on that evening. Eventually, all the participants were able to sign on to the bulletin board from home. In four of the schools the telecommunications were the link used to develop the projects. In the other schools, the telecommunications were used less frequently, and the messages were of a more personal nature.

September mini-conferences and site visits.

The mini-conferences served two purposes: a. They provided an intermediate milestone on project development with a reporting responsibility, and b. they gave the participants a chance to present their projects to peers in a "safe" environment. Before the first presentation, a few of the teachers were joking about being prepared, yet seemed ready to make their presentations. The presentations were quite varied, and the goals of each addressed different implementation issues. However the general consensus was that this event helped to synthesize the teams' efforts toward implementation and the November conference. They were seeing and sharing tangible results. The value of having intermediate milestones was confirmed.

Site Visits

The site visits showed many interesting features of the contexts where these projects were being implemented. Several deserve special mention:

1. At George Washington High, the project was presented at the District level and generated a great deal of interest. In fact, one other school in the district, Shallcross, wanted to get started this year with the Washington model.

2. At Dobbins, all of the teachers stayed after school to practice a software program for tenth graders. The teacher "computer expert" also presented procedures for helping students log on and practice safety measures in using computer hardware.

3. At Roxborough, staff development continued with the review and practice of software for math instruction. The department head, Sarah Poncz, remarked that even though the department had achieved its staff development goals, she plans to continue with the same program next year as well. "It takes more time than you think to practice computer assisted instruction."

4. Central's project focused on training 12 teachers to use the Apple computer as a measurement and recording device in lab experiments. Earl Feltyberger, Central's "teacher expert," introduced elementary procedures for interfacing with very simple exercises. Now teachers are starting to use prepackaged commercial software with some confidence and intermediate levels of expertise. At this school fewer teachers have time to continue with the project because of outside responsibilities, but those who are still involved are very enthusiastic. In one case a math teacher conferred with the "expert" in Physics about software for his algebra classes. This kind of interdisciplinary instructional planning has been happening in other schools as well.

Post-Institute Evaluation

Aspects of the institute were evaluated with a survey sent out to all the department heads in November. The results showed the following:

1. All respondents ranked the computer-assisted instruction as "most definitely or definitely worthwhile;

2. All respondents ranked the computer as an instructional tool "with a grade of "A" or "B".

Comments on "the aspects of the Institute liked best" included the opportunity to review software as a team, to share ideas with colleagues and to see other projects. Recommendations included increasing small group work and providing more time to process the instruction.

Phase 3 Implementation

Phase 3, initiated in the late fall of 1986, is designed primarily to extend the program to schools which have not previously participated in CAITP. The School District has identified science instruction as its priority area for the development of computer applications in instruction. Phase 3 consists of these main activities:

1. Completing of the installation of modems in the secondary school mathematics and science departments and strengthening instructional resources available through the PRISM electronic network;

2. Extending the competitive grant program to ten additional schools permitting them to acquire a MBL demonstration system based on an Apple IIGS system;

3. Providing teacher in-service training in the use of telecommunications and microcomputer-based laboratory applications in science instruction.

At the end of Phase 3, the CAITP program will have provided training to 38 department heads, more than 200 teachers, and directly impacted the instruction of more than 12,000 students. Thirty-eight schools will be linked into an electronic network for sharing instructional materials and information on science and mathematics instruction.

Conclusion

The strength of the program has been its collaborative nature. Funding for the activities has come from grants to Drexel University from PHEAA, Title II and to PRISM from the Ben Franklin Partnership through the Advanced Technology Center of Southeastern Pennsylvania. The School District has facilitated the in-service workshops, communications within the schools, and provided assistance in making hardware and software available to teachers. It is clear that no one of the partners could have accomplished these activities alone.

Evaluation methods for collaborative programs that emphasize formative aspects were found to be very useful. Sharing the results of the documentation also supported the collaborative efforts.

Although it is too early to say that the goal of increased effectiveness in instruction has been attained, there is sufficient anecdotal evidence to support a positive preliminary evaluation. The following comments are typical:

Department Head: "Before the summer workshop, I couldn't get x to touch a computer. Now he's on it all the time - before school, after school - I can't get on it myself, he's become so involved ..."

Computer Science Division Head: "They're knocking down our doors (to become part of the CAITP program). Before they didn't believe anything could happen. Now that they've seen some schools actually get some benefits, they can't wait to get in."

Teacher: "I love it. Could I keep the modem? I'm buying my own computer so I can stay tied to the network."

Perhaps the best evidence of goal attainments is found in the lessons and materials developed by the teacher teams. These will be presented by A/V slides and computer projection.

References

Beall, D. and Harty, H. (Summer, 1984). Inservice teacher reactions to implementing in elementary science and mathematics classes. The Journal of Computers in Math and Science Teaching, 34-38.

Cohen, M.R. (1979). Improving teachers' conceptions of computer-assisted instruction. Educational Technology, 19, 32-33.

Boyer, E.L. (1983). High School: A Report on Secondary Education in America. The Carnegie Foundation for the Advancement of Teaching. New York: Harper & Row, Publishers.

National Commission on Excellence in Education. (1983). A Nation at Risk: The Imperative for Education Reform. Washington D.C.

Sarson, S.B. (1971). The Culture of the School and the Problem of Change. Boston: Allyn & Bacon.

Sheingold, K., Kane, J., & Endreweit, M. Microcomputer use in Schools: Developing a Research Agenda. Harvard Educational Review, 53, 4, pp. 412-32.

	Science	Mathematics	Other
Apple-works	Database use	Algebraic solutions/ Graphing	Adminis. grade books Test scoring
Commercial	MBL interfacing Database use	Graphing Algebraic solutions	Adminis. grade books Telecomm.

Figure 1. Schematic of CAITP Hands-on Training

Key: 1=no,never; 2=almost never; 3=usually not; 4=sometimes; 5=usually yes; 6=yes, almost always; 7=always

Pretest: N=49; Posttest: N=40

ITEMS	pre	post
1. I am interested in school programs which have to do with microcomputer use.	6.0	6.1
2. Even with the extra work involved, I am willing to try microcomputers.	4.7	5.9
3. I keep abreast of the latest changes and developments of CAI and potential impact of computers.	3.8	4.5
4. I am willing to try CAI knowing the extra costs involved.	5.3	5.3
5. I feel fairly comfortable in reviewing new software for personal or school use.	4.5	5.5
6. I am very aware of specific needs of students regarding m/c instructional needs.	4.3	5.1
7. I like to experiment with instructional applications of microcomputers.	5.1	5.5
8. I tend to persevere with new ideas in matters of instructional development.	5.3	5.6
9. I tend to take a leadership role in m/c implementation efforts.	4.4	4.6

Figure 2. Mean Scores of Beall & Harty Survey

Schools	# part.	# times online	total # min.
CENTRAL	07	29	258
DOBBINS	05	20	158
G. WASHINGTON (2 departments)	12	51	587
GERMANTOWN	09	14	120
KENSINGTON	06	23	161
MASTBAUM	03	33	493
ROXBOROUGH	03	47	499
SAUL	05	17	254
UNIVERSITY CITY	11	58	776

Figure 3. Participants On-line Modem Times

UTILIZING VIDEODISCS IN EDUCATION:

TASK FORCE REPORT FROM THE 1ST NATIONAL VIDEODISC SYMPOSIUM FOR EDUCATION

Tom Burnett, Chair
Lincoln Public Schools
Lincoln, NE

ABSTRACT

Videodisc and other optical technologies, with or without the computer, have the capability of becoming exciting educational tools. Across the nation and at all levels and disciplines of education, educators are intrigued with the video and computer graphics capabilities of videodisc to create outstanding classroom presentations that meet different learning needs of students. The challenge is to capture and shape the direction of this fascinating technology to enhance education at the national level.

The purpose of the 1st National Videodisc Symposium for Education was to provide an interdisciplinary forum where educators, videodisc producers, and vendors could come together to share ideas, expertise and state-of-the-art videodisc programs. All symposium participants were involved in determining possible future directions for this exciting technology through the development of a "National Plan."

In order to facilitate the work of the symposium participants, a "task force" of national leaders in educational videodisc usage was assembled. These task force members created a framework by which information gathered at the symposium would be consolidated and disseminated. The task force also created mechanisms by which feedback could be received.

Panelists, each of whom is a task force member, will present information from their unique perspective regarding the symposium, the plan and planning process, and issues regarding the future utilization of videodisc and other optical technologies by educators.

PANELISTS

Richard Pollack
Minnesota Educational Computing Corporation
St. Paul, MN

Cynthia Char
Bank Street College of Education
New York, NY

James Mecklenberger
National School Boards Association
Washington, DC

Joe Clark
Videodiscovery

THE FUTURE OF EDUCATIONAL TECHNOLOGY IN THE HOME AND SCHOOL

Richard Wenn, Chair
EPIE Institute
Kenwood, CA

ABSTRACT

As educators, citizens, parents, and technologists, we are making value choices and decisions that affect the "technological impacts" on society and education as the Information Revolution proceeds. In this session the panelists work with the audience to consider future scenarios and how today's decisions, policies and actions may be creating those futures.

Part 1: The use of information technologies in schools homes, and communities is increasing, but how is it evolving? Beverly Hunter will consider how educational technology is likely to be used in schools in the future based on current trends and activities. Richard Wenn will present the educational potential of home information technologies, explaining that their future use is dependent on how the school system addresses home technology now. Ted Kahn will focus on the current and future use of technology in museums and libraries and on how these informal learning centers offer a metaphor for software design and learning centers of the future.

Part 2: Members of the audience ask questions, challenge the views of the presenters, and contribute their current experience to the shared visions being created.

Part 3: Through the use of "cardware," the audience will create future scenarios for productive uses of interactive technology. These designs will be discussed by both the audience and panel members.

PANELISTS

Beverly Hunter
Targeted Learning Corporation
Amissville, VA

Ted Kahn
Picodyne Corporation
Portola Valley, CA

EXPLORING NEW DIRECTIONS FOR COMPUTERS IN THE CURRICULUM

Henry F. Olds
Learningways, Inc.
Cambridge, MA

ABSTRACT

A New Model from Which to Build

The traditional education model for the study process is built on the assumption that you should learn all the prerequisite skills and knowledge before engaging in an effort to solve a complex problem. With the availability of the computer as a powerful and easy-to-apply tool, it becomes much more enticing to consider a completely opposite approach. If people are allowed to tackle a complex problem directly, they will quickly discover the skills and knowledge they need to reach a solution. They are then motivated to gain the skills and get that knowledge.

This model for a new curriculum suggests that much of the learning that goes on in schools ought to be an outgrowth of students' wanting to know about something, wanting to gain expertise, and wanting to become good learners. With the help of new computer tools, we may be able to create the kinds of learning contexts (problem environments) that will bring about the intellectual engagement this model requires.

Furthermore, since the computer is primarily a visual tool, the problem environments will be full of visual images. The solutions to problems will come about through the manipulation of visual images, and the solutions themselves will be presented in part in visual images.

Ultimately, using these new tools to solve problems will alter our way of seeing those problems. It is a truth of human nature that because people are tool-using and tool-dependent creatures, their capacity to understand their world is closely related to the tools they have at hand. New tools make possible new ways to know about things and, ultimately, new understanding. Problems that once seemed complicated and difficult may soon seem simple and easy, and new aspects of how to arrive at satisfying solutions will be learned.

APPLE CLASSROOM OF TOMORROW

Keith Yocam
Apple Computer, Inc.
Cupertino, CA

ABSTRACT

The Apple Classroom of Tomorrow Project (ACOT) is a working partnership among Apple Computer, Inc. and seven public school classrooms in which each student and each teacher has one computer at school and another at home. These seven settings are "living laboratories," bases for a long-range, action research and applied development agenda.

Apple provides hardware; full-time site coordinators; technical assistance; introductions to software developers (who have donated more than $170,000 worth of software to the sites); ongoing support in the form of a telecommunications network among the sites, monthly visits from ACOT staff, and bi-annual ACOT participant conferences; management to hold it all together; and leadership to keep it moving forward. The schools contribute classrooms, wiring and furniture; able teachers who dedicate countless hours to the project's success; and district support to protect and nurture the ACOT educational frontiers.

The ACOT objectives are:

- To capture critical knowledge about the integration of computers into instruction, across the curricula, in the classroom and the home;
- To identify and solve many of the problems of computer implementation in classrooms;
- To discover new and productive avenues for hardware and software development; and
- To disseminate program findings to many stakeholders in the educational process: parents, policy makers, educators, and hardware and software developers.

The mission of Apple Computer is to create tools for the mind and to place them into the hands of every person. ACOT brings that vision to life in selected public school classrooms across the United States by empowering all teachers and students in those classrooms with computers as teaching and learning tools.

ACOT has in its first year installed and brought to full operation seven classroom sites around the United States. The project has begun to yield useful information and has garnered the attention and respect of an ever-widening circle of practicing educators. The benefits of the work in these classrooms will become even more apparent during the next year of operations.

Apple Computer Inc.'s sponsorship of ACOT demonstrates to the public the company's long-term dedication to the K-12 education community. The most important fact about ACOT is its very existence. It is a visible reminder of Apple Computer's commitment to leadership in education.

PANELISTS

Thelma Scott
Lester Demonstration School
Memphis, TN

Jane Pratt
Columbus Public Schools
Columbus, OH

TOOLS FOR THEORY BUILDING AND PROBLEM SOLVING

Tim Barclay
Technical Education Research Centers (TERC)
1695 Massachusetts Avenue
Cambridge, MA 02138

Nancy Roberts
Lesley College, Graduate School of Education
29 Everett Street
Cambridge, MA 02138

Abstract

Focusing on curriculum materials and software for high school students, this project is combining microcomputer-based data gathering instruments with model building from the system dynamics perspective. The microcomputer laboratory aids students in collecting real-time data in such areas as heat, light, motion, and water movement. Model building helps students develop and test theories about their observations. An integrated software environment is being developed that allows data to be input directly into the model from the probes or the keyboard. In addition, the software includes an icon based modeling language, an algebraic function grapher, and a simplified spreadsheet to provide students with several different approaches to model building and simulation. Each of the components of the software output data in the form of tables and graphs.

The first trial of curriculum materials was conducted during the summer of 1986. During the 1986-87 academic year the curriculum materials and software were refined and again classroom tested. The 1987-88 academic year will provide an opportunity for more extensive trials in regular classroom settings. The research goal of the project is to better understand high school students' ability to construct computer models of natural and social science phenomena without the prerequisite of calculus. In addition the project is:

- studying differences in problem-solving strategies of males and females;
- attempting to identify the elements that support a successful problem-solving environment such as interactive icons, real-time data collection, and real-world problem solving; and,
- looking at students' ability to transfer concepts of basic structure between problem areas and disciplines.

Project Overview

This project builds on work underway at Technical Education Research Centers (TERC) for the past three years. The Microcomputer Based Laboratory project (MBL) demonstrated that middle school students can intuitively interpret the graphs of data they collect in real-time with the aid of a computer. Sophisticated concepts such as heat and temperature, and velocity and acceleration, were understood by these middle school children. Looking for ways to extend students' problem-solving skills by taking advantage of their ability to understand graphs led to our current project.

Hardware, software, and curriculum are being developed for integration into high school science, social science, and applied mathematics courses. The curriculum packages deal with subject matter normally found in these high school courses. The hardware allows for real-time data gathering using a Macintosh computer while the software provides an integrated tool for problem solving. The software has four parts, all outputing data in tabular and graphical form:

1. data from microcomputer probes;
2. an icon based model-building language similar to STELLA;
3. a simplified spreadsheet;
4. an interactive algebraic function grapher.

The thrust of this project is to give students strategies and tools for solving complex problems. The curricular materials present students with realist natural or social science problems. The computer based tools aid students in developing hypotheses, collecting data, and testing their hypotheses using simulation models validated, when possible, with data collected by the student. Structuring a data collecting experiment focuses on the problem-solving strategy of dividing a problem into small, manageable parts. Theory building with the aid of modeling and simulation requires a holistic and integrated approach to problem-solving. Students are given experience with both strategies so as to develop a facility with both and a sense of when each is appropriate.

The curriculum is being designed to allow students to apply their intuitive mathematical knowledge without the prerequisite of formal study. The algebraic function grapher is included to further develop mathematical understandings through interactive software.

Transfer of learning between disciplines is another focus of the project. Taking the system dynamics approach to model building, the students are introduced to a uniform way of expressing the underlying structure of problems amenable to analysis by identifying the causal links and feedback loops that account for the observed behavior. (Forrester) Such problems are found in all the natural and social sciences.

All the materials being developed emphasize the multiple representations of abstract ideas. Theories about behavior, such as exponential growth, can be expressed using icons as shown in Figure 1, as equations as shown in Figure 2, in a spreadsheet format, as an algebraic function, or using probes (see Summer Experience below). Each one of these pieces of the software output the results of the model as a graph or table as shown in Figures 3 and 4. The icons, equations, or probe experiments can be manipulated by the students to ask *what if* questions that give immediate results through graphical output.

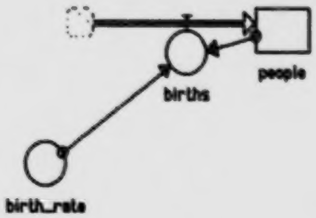


Figure 1. Population Growth Model as Icons

□ **people = people + births**
INIT(people) = 150
○ **births = people*birth_rate**
○ **birth_rate = .04**

Figure 2. Equations for Population Growth Model

Time	people	births
0.0	150.000	6.000
10.000	222.892	8.916
20.000	331.206	13.248
30.000	492.155	19.686
40.000	731.316	29.253
50.000	1086.697	43.468
60.000	1614.774	64.591
70.000	2399.470	95.979
80.000	3565.486	142.619

Figure 4. Tabular Output of Population Growth Model

Research Questions

The curriculum and software being developed for this project allows the researching of several important questions on the impact of computers on teaching and learning. Most critical, from the perspective of the project staff, is the potential for providing students with an intuitive sense of calculus and an ability to use the analytic power of this branch of mathematics without necessitating its formal study. Work by Roberts and others over the past 10 years has provided evidence to suggest this is possible.

A second area is documenting the results of giving students the ability to build and and use simulation models for social and natural science investigation. Providing high school students with these powerful tools creates a truely interactive problem-solving environment. For some time now educators have been recommending interactive environments as the most natural way to provide students with meaningful problem-solving skills. In addition, model-building requires synthesis thinking creating a natural way to engage students in higher-level thinking.

Another area of general interest, especially to science, mathematics, and computer oriented researchers, is differences in learning styles, achievement rates, and abilities of male and female students. This project hopes to contribute more data on this topic. Our main hypothesis, that both sexes can learn equally well given an appropriate curriculum, is based on developing multiple representations of ideas, especially mathematical ideas such as *function* and *change over time*. The curriculum presents each concept in graphic, iconic, number, and word representation. In addition, the tools being developed have a high degree of interactiveness both physical and mental, to provide for a diversity of learning styles.

One area of research, only recently possible due to advances in microcomputers, is studying the impact of interactive iconic environments for teaching difficult concepts. Our hypothesis is that these dynamic, interactive pictures of abstract concepts will reach a group of students traditionally lost at this level. The MBL project has produced some evidence that middle school students, involved in producing on-line graphs, are able to understand distinctions between heat and temperature, velocity and acceleration in relatively short periods of times. We are extending this pedagogy to other environments that present ideas as pictures.

Finally, the true pedagogic rewards, recognized by researchers such as Bruner for quite some time, is the ability of students to identify the underlying structure of a problem and transfer this understanding to other problem areas. To facilitate this happening the curriculum being developed encompasses several disciplines. For example, units looking at the cooling curve in physics, exponential growth of terrorism in social studies, and the exponential decline of endangered species in ecology are included. Modeling the same phenomenon in several different disciplines will hopefully strengthen the perceived value to problem-solving skills of identifying the underlying structures that cause the observed behaviors.

Prepilot Summer 1986 Experience

During the summer of 1986, we ran a summer school program as a chance to try out many of the modeling curriculum ideas we are considering. Sixteen incoming tenth and eleventh graders came from 9 am to 12 noon each day for four weeks. We viewed this as an opportunity to test the feasibility of our ideas. None of the students had more than one year of algera. Those that had a second year of high school mathematics had taken geometry. The students' algebra knowledge, as measured by a standardized algebra test given on the first day of class, was quite spread.

During the first week students became familiar with the Macintosh computer, with using probes to collect data and generate real-time graphs, and with graphs and functions and the ways in which graphs can be transformed. The functions and graphs activities use a fast graphing utility developed for the Macintosh. This allows the user to change, using the scroll bar, any of the coefficients and constants of an algebraic function and get an immediate replot of the graph. In addition to providing a manipulative way to learn how to transform graphs on the coordinate grid, this adjusting capability becomes a powerful modeling tool when trying to fit a graph to a data plot.

During the second week the focus was on growth functions as exemplified by bank balances and populations. Students modeled these phenomena using a physical model involving the movement of water into and out of beakers, which we call Leaky Buckets. Since we interface these buckets to the Macintosh, it is possible to generate real-time graphs of the water level as a function of time. Linear growth is modelled by adding a fixed amount of water to the beaker at each time increment; exponential growth by adding an amount proportional to the level of water in the bucket. These same situations are then modelled using a spreadsheet and iterative functions to generate the growth data.

Introduced during the third and fourth weeks were causal-loop diagrams as a way to think about systems involving feedback, and the icon modeling software for building dynamic models. Two probe activities were also included, this time with the opportunity to make a model of the system and then test it against real data collected with a probe. The activities involved a cooling curve experiment and an electrical capacitor discharge experiment. Several problems from the social sciences, such as urban growth and terrorism, as well as more population problems were also included.

Throughout the four weeks there were really two goals: the first more skill oriented, involving learning to use the Macintosh and the probes and the software tools; the second, more concept and process oriented, learning to use systems thinking and modeling as a way to understand complex systems and to make predictions about behavior over time. The first goal was easy. The students had little trouble mastering all the hardware and software. The second goal was harder to attain and harder to evaluate. What was exciting was student success in working with the problems and challenges that were presented; what was disappointing was the minimal carry-over of systems thinking and problem solving from one activity to the next. This latter is perhaps not surprising, given the short time of four weeks and the trial nature of the acitivites. On the positive side, all the students had significant learning, as demonstrated in the analysis of the pre-post test scores, and all students, regardless of their mathematical backgrounds, learned proportionate amounts. Some comments made by the students at the end of the summer were:

"I'd recommend this to anyone. I don't think it matters whether you like math or not; it's just something that helps you and gives you information."

"We did things in different fields like accounting, we did savings accounts; we used things like leaky buckets, we did trigonometry. We just went into a lot of different fields, and it was really fun and I had a great time."

And by the teacher:

"We as a society need to learn how to make better long-range solutions. If we are going to develop the future leaders of this country, we have to give them the tools for doing this. Mathematical modeling is one tool that really looks like it is going to work, and I think students should have it in their hands."

"Kids love it because now they are learning about problems by playing with them. They are not learning about equations and ways of manipulating equations; they are learning about real-life situations and how things change in the real world. They also have a great deal of control over their learning because they can make their own assumptions and see what happens. A very exciting classroom situation is created when you start using these materials."

Goals and timeline for 1987-88

Informal pilot testing is scheduled for two sites in the spring of 1987. At this time only partial units and some software will be available so the testing will be as enrichment for current units. We are planning to try small pieces of material in a social studies, physics, biology, and mathematics class. For the 1987-88 academic year we plan to develop totally integrated units that will reflect our spring pilot findings and change the traditional curriculum to integrate the new teaching strategies and computer-based tools. We also hope to develop and pilot a one term course in system thinking and model building. A more formal data collection will occur.

Summary

The products at the completion of this project will include:

1. An integrated tool software package that outputs data in graphic and tabular form from probes, algebraic functions, a simplified spreadsheet, and a modeling program;
2. Five units on general aspects of the curriculum, system thinking, and problem-solving including an introduction to:
    - modeling;
    - causal-loop diagramming;
    - the integrated software package;
    - system thinking including the Leaky Buckets physical apparatus to introduce the concepts of levels, rates, and feedback; and
    - the use of probes and reading graphs through the MBL unit on motion.
3. Written materials consisting of student activities and teacher guides for three to five content specific units in the areas of physics, biology, mathematics, and social studies. In addition, a one term course on system thinking and model building will be developed.
4. Evaluations and research results based on the pilot testing of the curriculum materials.

References

Bruner, J., *The Process of Education* . New York: Vintage Books, 1963.

Forrester, J. W., *Principles of Systems* . Cambridge, MA: MIT Press, 1968.

Roberts, N., "Teaching Dynamic Feedback Systems Thinking", *Management Science*, Vol. 24, No. 8, April 1978.

STELLA.. Hanover, NH: High-Performance Systems, Inc., 1985.

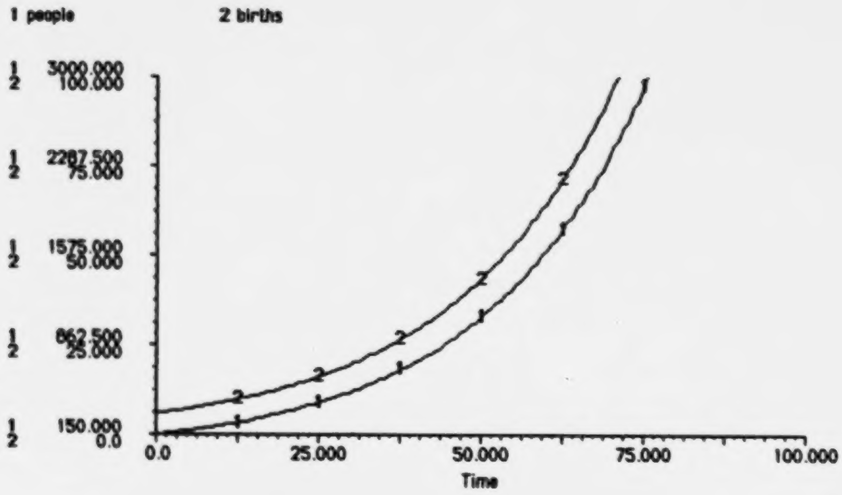


Figure 3. Graphical Output of Population Growth Model

CONDUCTING A CLASS IN HYPOTHESIS TESTING WITH A SINGLE COMPUTER

Jon B. Latson
The Wheeler Computer Center
The Wheeler School
Providence, RI 02906

A great deal of emphasis in educational computing is placed on one-student-one-computer activities. But what about a single computer as an instructional tool for the teacher standing in front of the class? How can the use of a computer go beyond the "electronic blackboard" use of demonstration programs to change the role of the teacher in the classroom? One way, being examined by the Rhode Island / Brown Technology Consortium (RI/BTC), is to simulate the "instructor" with the computer, allowing the teacher's role to change from lecturer and conclusion drawer to initiator and facilitator in the context of a high school biology curriculum. RI/BTC is a research group, funded by the Educational Technology Consortium at the Harvard School of Education, consisting of a biologist and an educational specialist, a science curriculum and instruction specialist, and a programming consultant, supported by teachers in several local high schools who have offered their insights into planning the instructional unit, assisted in its preparation, and piloted the computer-driven unit in their science classes.

Our work has been designed to address two concerns. One concern is the degree to which students do little more than memorize facts, laws, and principles. Textbooks and lectures are *descriptive* in their approach rather than *experimental.* In theory, class laboratory experiments are intended to provide opportunities to explore problems, manipulate variables, and individually draw meaningful conclusions. In practice, however, the experiments become simply demonstrations of facts already known to the student. Furthermore, laboratory work is frequently unsuccessful because of time constraints or student confusion and error. Students often do not go through the complete process intended in the experiment, spend little time analyzing what they are doing, and are afraid to share hypotheses. In many cases, insecure about their own results, students want the final, "correct" answer.

Another concern involves the role of the teacher in the science classroom. All too often he/she is the primary focus of the educational activity. Using the declarative tools of lecture and textbook, the teacher becomes the "expert" intent upon the subject matter, presenting, explaining, and exemplifying those scientific concepts and principles that were originally derived by exploration, experimentation, and speculation.

Our approach has been to use a single computer with a large screen and video projector as the focus of the class running software that engages students in the process of examining phenomena related to genetics. The computer directs the class by presenting a certain situation; the class selects experiments to test the situation, gathers data, and discusses the ongoing investigation at the program's prompting to determine what to do next.

Given such a scenario, we have asked the following research questions:

1. Will science teachers be receptive to a new pedagogical style in which the teaching pace is partially set by a computer?

2. Will receptive teachers perceive their role in the classroom differently, and will they be able to modify their teaching style?

3. Will computer-guided activities in front of a class be an effective focus of attention and address our underlying concerns?

4. Will students engage in inductive reasoning and establish logical connections while testing hypotheses?

DESCRIPTION OF SOFTWARE AND MATERIALS

The software, written specifically for this project, examines the distinction between people who can taste Phenylthiourea and those who do not taste that chemical, and consists of four programs which are administered in order: "Filter Paper Experiment," "Taster Marriages," "The Islands," and "Two Jars." The program design uses graphics, animation, and sound to attract student attention.

"Filter Paper Experiment" opens with a sequence of directions to the teacher to conduct an experiment by tasting untreated, then treated, filter paper. In a typical class, several students will not taste any difference. The teacher enters the number of Tasters and Non-Tasters and the percents of the total number of students are displayed. Students record this information on their Data Sheets and consider the question: "Are these data typical for other classes?". The next screen displays data of other classes (each of 20 students), and the first hypothesis is presented: "The percentage of T and NT is the same for any group." Students are asked to consider the question that will appear after every hypothesis: "Are the data consistent with the hypothesis?". Since the data show a typical spread of percents for groups of 20, students will answer No or be asked to think again. A second hypothesis is presented still showing the same data and the question: "There are more Tasters than Non-Tasters for any group."

The answer here will be a Yes. Students are prompted to consider what other tests might be required to accept hypothesis II and are provided with a menu to see: 1) other groups of 20, 2) smaller groups, 3) larger populations. Hypothesis II will be disconfirmed after collecting sufficient data about small groups. After several selections from the menu, students are allowed to consider a third hypothesis: "The percentage of T and NT is the same for all LARGE groups." This hypothesis is confirmed after looking at all the data about large groups (Non-Tasters comprise approximately 25% of a population).

"Taster Marriages" opens with the hypothesis: "Tasting ability is inherited." Students are presented with five males and five females, each of whom is labeled Taster or Non-Taster. Students may "marry" any pair and observe the Taster/Non-Taster breakdown of their progeny (20 children are observed after each marriage). Students continue to gather data until they feel they have tested the opening hypothesis sufficiently. Then they must respond Yes or No to three statements: 1) Two NT parents have only NT children, 2) Two T parents have only T children, 3) One T parent and one NT parent have only T children. Because Tasters may carry an NT gene, statements #2 and #3 are correctly answered "No." After responding to the statements, students may return to do more experimenting to determine that phenomenon.

"The Islands" opens with a picture of three tropical islands labeled Taster, Non-Taster, and Meeting. A story is told about how all the inhabitants of Taster island have always been Tasters and all inhabitants of Non-Taster island have been Non-Tasters. One day natives of each island go to Meeting island where each marries someone from the other island. Their progeny, the first Meeting island natives, are all Tasters. The three statements from "Taster Marriages" are posed again, and, in light of the data concerning the progeny on Meeting island so far, students are asked to respond to each statement. The Meeting island natives grow up and marry one another, and data is collected on their children: out of 4000 children of Meeting island natives, 3000 are Tasters and 1000 are Non-Tasters. Students are again asked to consider the three statements, in light of the most recent data.

The next frame of the program invites students to "marry" any combination of inhabitants on Meeting island for whom they do not already have data concerning progeny. When students feel they have gathered sufficient information, they are asked to consider three hypotheses and fit every possible datum collected from the three islands to confirm or disconfirm the hypothesis. Hypothesis I: Non-Tasters pass a factor to all their progeny which prohibits tasting. Hypothesis II: Tasters pass a factor to all progeny. Non-Tasters lack this factor. Hypothesis III: Each offspring receives a factor from each parent; offspring with one taste factor are tasters. Only Hypothesis III satisfies the entire set of data.

"Two Jars" opens with a recapitulation of Hypothesis III and discusses its ramifications by presenting the model of each parent as a jar containing balls that are labeled T or N. Beginning with T's and N's in each jar, students are asked to predict the outcome of each of the four possible combinations of factors (balls) from the mother and father.

Next, students are offered a menu of selections in which they may observe data concerning single or multiple offspring for parents whose type the students determine (either all N's, all T's, or N's and T's). A third choice is to guess the mother's type by selecting the father's type and observing a sufficient number of offspring. In the course of guessing the mother, students are asked to consider the effectiveness of each possible choice for the father's type in determining the mother's type.

Beyond the software, instructional materials include a supply of red (untreated) and blue (treated) filter paper for the first day's experiment and Data Sheets, forms created to correspond to the data presentation on the screen to facilitate the transcription by each student of the results of various simulated experiments during the unit. These tally sheets allow students to refer to previous information without calling up the screen display again and to keep a record of the results for the homework assignments and the following day's activity. Homework assignments are given after each of the first three lessons.

The software was piloted in two school settings, first at an independent, coeducational, college preparatory day school, and then at a public, college preparatory school. The classes were observed and videotaped. Following the pilots, and with feedback from biology teachers who had only previewed the software, revisions were made to the software and materials for the formal experiment.

THE EXPERIMENT

A six-hour orientation session was held for the six teachers taking part in the experiment who had not been involved with our initial pilots. Background information on RI/BTC was provided and a videotape of selected portions of the pilots was presented to acclimate the teachers to the intended pedagogical techniques. Each teacher received a packet of instructional materials, a storyboard outline of the software, and the pre-tests and post tests. A period of time was allocated to familiarize the teachers with the content of the software. The session ended with a wrap-up and a review of the intended behavior of the teacher as facilitator.

The first segment of the experiment involved three teachers at an inner-city school with a student body reflecting a racial and ethnic mix of Cambodians, Hispanics, Blacks and Caucasians. Five sections, a total of 115 9th grade students, were involved, ranging in ability level from high to low middle. The second segment of the experiment involved three teachers at a suburban high school. Each teacher had two 10th grade sections of Biology I. Four sections were of average ability, and two sections were for low ability students (6th grade reading level) placed in a special biology program. All the sections had previously received instruction in genetics. Observations were conducted and video tapes made of most of the classes.

DATA COLLECTION

We have collected data from eight teachers in fifteen classes to assist us in the further development of project materials and to provide information useful in answering the research questions. Because the project was a short term (3-4 days) intervention in the classroom, there was little reason to expect major changes in students' skills in hypothesis testing. However, it was expected that the project should have a perceptible impact on the teachers and students and an observable impact on the conduct of the class. We asked ourselves: 1) What happens in the classroom? 2) What is happening to students? and 3) What is happening to teachers? In order to collect data relevant to these questions, the following techniques were used: classroom observations, teacher pre-/post questionnaires, teacher debriefings, previewing teachers' responses, student post questionnaires, student pre-/post tests, and video tapes.

RESEARCH FINDINGS

The first research question asks if teachers would be receptive to a new pedagogical style in which the teaching pace was partially set by the materials. Teachers who used the materials and teachers who previewed the materials were two groups who helped us answer this question.

All of the teachers who used the materials stated during the teacher debriefings that they would use the technique again. The pilot teachers indicated that they did not readily adjust to using the materials, and that teaching a second class in this manner was much more comfortable than their first class. A few teachers indicated that they might modify the classroom setting from the way they used the materials in the trial.

Most of the teachers who previewed the materials indicated that they would use the materials in their classes. These teachers found the materials appealing because they required the students to engage in higher level thinking and to use the scientific method. About one in eight the teachers previewing the materials expressed reluctance to use the materials in their classroom. This reluctance was often tied to their preference for other materials or teaching strategies.

While teachers were found to be receptive to a science teaching approach in which the pace is partially set by the computer, several new parameters of concern developed during the experiment. One concern was that this mode of instruction may have extended for too long. Another concern was the limited number of entry/exit points in the software. Class periods were of different lengths during the pilots. Teachers found it difficult to determine appropriate ending points for a lesson to match the class period. It appears that teachers are willing to use this instructional method occasionally, but not constantly. It also appears that teachers want the structure of the material flexible enough to adapt to their instructional setting.

The second research question asks if teachers would perceive their role in the classroom differently and if they would modify their teaching style. Data from the teacher questionnaires, teacher debriefings, classroom observations and the videotapes provide information related to this question.

Teachers perceived that using the computer and the software in the classroom altered their teaching style, that it was new and different. They indicated that they could attend to different classroom activities because the primary script for the lesson was programmed into the computer. Not all of the impact on teaching style was due to the structure of the lesson. Mechanical placement of the projector in the classroom, the darkening of the classroom, and the control of the computer were issues related to the equipment that teachers felt affected their teaching style.

A major effort went into examining teacher behavior in the classroom. The videotape record of pilot classes allowed for more careful examination of teacher behavior. One area examined was the cognitive level of questions used by teachers. Teachers asked from 35% to 51% high level inquiry (as categorized by Ladd and Andersen, 1970) questions. Research review by Balzer (1973) and Gall (1984) indicate that Biology teachers normally do not exceed 20-25% high level inquiry questions. We feel confident that the materials have helped to promote higher level questioning in the classroom.

The fact that teachers used numerous high level questions does not in itself indicate that our use of technology has been successful in altering the role of the teacher to that of a facilitator. Observers felt that few, if any, teachers used the computer to enable themselves to assume a facilitating role in the classroom. In spite of the fact that three of the teachers had been project staff members and all teachers had undergone training, they continued to overwhelm students with their talk and questions. They were uncomfortable giving students "wait" or "think" time or with exploring students' ideas. Teachers asked an average of 235 questions in 150-160 minutes of instruction. We believe this rapid rate of questioning reflects ingrained teacher behavior. While previous research (Evans, 1973) has indicated that instructional behavior is resistant to modification, we had assumed that the computer software might instigate a change in the teacher role. The self-reports of teachers and the analysis of the teacher questioning style indicate that some shift in teacher role has occurred, but not nearly to the degree we had projected.

The third research question asks if computer-guided activities in front of a class would be an effective focus of attention and address our underlying concerns. Teacher reports, videotape reviews, and classroom observations indicate that the computer can be used in the role envisioned.

Strengths and weaknesses associated with this method are becoming clearer. The computer can be the focus of attention. Students can attend to the computer presentations, they can follow its instructions, and they can respect its judgments about their responses. Teachers, however, were not immediately comfortable with the computer as the center of attention. During the debriefings, teachers noted that it was difficult to lead a class discussion when the students' eyes were directed at the projection screen. The mode of instruction appears to promote student discussion and consideration of diverse views. Student questionnaire data indicated that this instruction differed from the normal science instruction they received in the amount of group discussion that occurred.

On the other hand, the computer as a focus of attention appears to lose its effectiveness over time. Continued use of this instructional methodology can lead to student boredom. To the student, each lesson appears the same. The computer projects an image on the screen in a darkened room, the class discusses an item, a choice is made, and the cycle is repeated. Teachers indicated that clever graphics, especially those with animation, can help to refocus student attention on the screen. Even so, the sequence of lessons showed an increasing number of students who either lost attention or became marginally attentive to the lesson. While contextual factors may have been partially responsible for this (students knew they were not graded; students may have previously studied the content), the lesson structure may share responsibility. Much of the logic of the lessons develops in a long linear sequence. If a student misses or fails to understand an intermediate step in this linear sequence, the student may elect not to participate in the remainder of the lesson. It was our hope that giving part of the instructional role to the computer would free the teacher to attend to the issue of student participation. Little evidence of the teacher accepting this role was found.

The computer, rather than the teacher, dominated the structure, though not necessarily the delivery, of the lesson. The computer had the obvious control over the sequence of the material. Often teachers were heard to apologize for not "following" the computer script, e.g. asking a question one frame before the computer asked it. Lessons did not begin with the usual overview provided by the teacher. Moreover, the usual summary or closure activity presented by the teacher in most lessons formats was omitted from our lessons. Observations indicated that students were often following the lesson without knowing what their task was. At this stage we are considering which aspects of the whole group instruction are best left to the teacher.

The fourth research question asks if students would engage in inductive reasoning and develop skills in hypothesis testing. The long term purpose of our project is to develop student skills in this area. Throughout the instructional materials students were presented with hypotheses and asked if data support them or asked what data should be collected. We have used data from classroom observations to determine if students are developing and using hypothesis testing skills. The classroom observations have provided us with mixed feelings about our success in promoting hypothesis testing skills. Often we observed instructional sequences in which neither the teacher nor the students appear to "get the point." In other cases we have seen students do a superb job of recognizing the relationship of a set of data to an hypothesis. Overall, classroom observations indicate that students do not gain the hypothesis testing skills in this one set of lessons. Our current thinking is that students will have to encounter several examples of hypothesis testing throughout a biology course in order to develop these skills.

CONCLUSIONS AND ISSUES

Concerning students engaging in scientific reasoning, first, we can say that, in the course of the experimental unit, we observed that students did practice hypothesis testing. Second, not all students could do this, and those who could not tended to drop out of the lesson. Third, we can conclude that extended practice is needed to accomplish a significant change in a student's scientific reasoning. Finally, from our own experience and the existing research, we know it is not easy to measure a change in student skill in this area.

Two issues must be addressed in our next steps. First, the duration of the lesson may be critical to keeping students involved and to maintaining a freshness in the method of instruction. Second, more separation is needed between thinking about the lesson context, in this case genetics, and thinking about the scientific process. The metacognitive aspects of this unit, thinking about what makes an hypothesis consistent or inconsistent, may very well be overshadowed by the specificity of the questions asked by the computer concerning a genetic phenomenon. This second issue is bound up in the prevalent concern both students and teachers have about the immediate content to be understood, as opposed to the process of thinking about the content.

For the second concern, that of changing the role of the teacher in the classroom, we know that adopting the role of facilitator was extraordinarily difficult for all the teachers who tried it. Some teachers perhaps did not really understand what we intended, but even for those who were fully aware of the intent, putting that mode of instruction into practice proved terribly demanding and at times uncomfortable. We know that part of the normal instructional control attributable to the teacher in a class activity was transferred to the computer used as we have designed it. Teachers noted a loss in control both in structuring the class and in managing student behavior.

We know that the teacher's questioning of students during the classes included a high percentage of high-level inquiry questions. This result may have been a direct result of the presence of the technology and the instructional intervention of the software, or it may have been the result of heightened awareness on the part of teachers to ask higher level questions while they were being observed. In either case, we failed to observe other anticipated teacher behaviors associated with the role of the facilitator, such as the presence of wait time after questions were asked and the incorporation of student ideas into the selection of choices from the software.

Still an issue is the means by which we can effectively alter teacher behavior. One possible avenue of exploration includes a more comprehensive teacher training module that provides clearer models for teachers to follow. Another avenue would be to provide more elaborate materials to structure the lesson and the instructional methods, or to build into the software more guidance for the teacher. This issue leads to two further research questions. First, which teacher behaviors are most desirable and what changes need to occur to facilitate the development of scientific reasoning among students? Second, to what extent should the computer have an instructional role in the classroom, given the relative inflexibility and "blindness" of the current technology to react to individual class needs, and which instructional aspects are best left to the teacher, given the inevitable diversity of teacher styles?

REFERENCES

Balzer, Le Von. "Science Instructional Behavior: Instrument Development and Basic Data Trends," *A Review of Research on Teacher Behavior* (Balzer, Evans, Blossen). Association for the Education of Teachers in Science. Columbus, Ohio (1973), pp. 1 - 126.

Evans, Thomas P. "Research of Teaching Involving the Systematic Observation of Classroom Behavior" (1961 - 1971), *A Review of Research on Teacher Behavior* (Balzer, Evans, Blossen). Association for the Education of Teachers in Science. Columbus, Ohio (1973), pp. 127 - 252.

Gall Meredith. "Synthesis of Research of Teachers' Questioning," *Educational Leadership* Vol.42, No. 3, 1984, pp. 40 - 47.

Ladd, G. T. and Andersen, H. O. "Determining the Level of Inquiry in Teachers' Questions," *Journal of Research in Science Teaching*, Vol. 7, No. 4, 1970, pp. 395 - 400.

181

WCU MicroNet
Update on a Computer Network

Linda J. Perry, Patricia G. Johnson, and J. Roger Bacon

WCU MicroNet
Western Carolina University
Cullowhee, NC 28723
(704) 227-7633

Abstract

WCU MicroNet is a statewide computer-based science network providing communication, enrichment activities, and resource sharing among approximately 65 secondary schools, Western Carolina University faculty and library, N.C. Wildlife, and other agencies. Services on-line include electronic mail, a conferencing/bulletin board system, several databases, and graduate level science courses. These and other features are available to subscribers 24 hours/day, seven days/week. The network concept and software are generic and may be used for disciplines other than science.

History

Western Carolina University is located in the western tip of North Carolina, which is predominantly rural and mountainous. Science teachers are relatively isolated from one another and have had little contact with research scientists. The WCU MicroNet Project was conceived early in 1981 by Dr. Roger Lumb and Dr. John J. Manock of the WCU science faculty to make university resources more accessible to pre-college classrooms and to promote effective resource sharing and idea exchange among secondary school science teachers, their students, and university faculty.

In the fall of 1982 MicroNet went on-line using the university's VAX 11-780 computer and boasting one participating school. By the end of the first year, the total number of on-line schools had grown to approximately twenty. The number of subscribers has stabilized at about 65 schools plus numerous WCU faculty members and contributing agencies. Since we first reported on MicroNet at the 1984 NECC meeting, we have solved most of the problems indicated at that time and have developed a stand-alone system complete with our own operating software. The system has been shaped by teacher needs and designed specifically as a model for educational networking, with applications beyond the field of science. In view of the number of inquiries we are receiving from people interested in starting networks, we feel a formal progress report presented through NECC would be useful.

System Description

WCU MicroNet is currently running on a single Tandy 6000 microcomputer with one megabyte of random access memory and 15 megabytes of hard disk storage space. Three WATS telephone lines and one local line (for WCU faculty and staff and some local schools) connect to the computer via modems. Three on-site terminals and one link to the campus VAX 780 computer within the MicroNet office bring the total number of usable ports to eight.

The operating system in use is Xenix, version 3.1. Rather than sending users directly to the user-unfriendly Xenix shell, the system presents them with a menu driven shell. The menu routine gives the user a choice of three levels. Beginners are guided through their initial and all successive logon sessions by menus and optional help messages, while the advanced computer user may choose to enter Xenix directly without any form of help. A command level is available for the intermediate user who has memorized the functions and does not wish to be prompted with menus.

The electronic mail system is that which is supplied with Xenix. Other software used (conferencing system, database, tutor/testmaker, course exam, and menus) was developed in-house and was written in the C language. Since the C language is highly portable, programs should be fairly easy to adapt to other systems. (We had temporary use of an AT&T 3B2/300 last summer and successfully transported programs to that system.)

Features

Individual School Accounts. Upon approval, each school receives an individual school account. The account consists of a username, password, directory, and mailbox. The username is usually a shortened version of the school name, to make recognition of mailboxes easier. A password (necessary to gain access to MicroNet) is entrusted to a science teacher who serves as official contact person and determines which students will be permitted on-line. Each account directory acts as a storage place--a physical area on the computer hard disk which is maintained by the account user. Teachers and students can save mail messages, text, or program listings in their own school directories which can then be accessed by others on-line.

Electronic Mail. Each account, when created, is assigned an electronic mailbox to which others may send private messages readable only by the recipient. The mail feature is often used for communication between individuals at a school and a WCU faculty member; it is also used by students to answer the weekly quiz competition questions.

Conferencing/Bulletin Boards. This feature facilitates interaction among all of the MicroNet subscribers on a public 'forum' level. Students, teachers, university faculty, and any other user may subscribe to and post messages on any of the available conferences. Replies to messages may be made privately through electronic mail or publicly through the conferencing system. Conferences currently active include "announce" (any science-related announcements); "micronet" (pertaining to the operation of the MicroNet Project as a whole); "quiz" (weekly quiz questions posted in biology, chemistry, earth science, and physics); "micro.use" (the use of microcomputers in the science classroom); "nat.sci" (discussion of natural science phenomena); "earthsci" (discussion of related topics especially for new earth science teachers); "robotics" (robots, purchasing information, instruction, and uses); and "bio693" (course discussion topics). Conferences are created or deleted in response to user interest. The user may choose to read only new messages since the program keeps track of which conferences each user subscribes to and how many messages the user has read within each conference. Articles may be scanned (by a keyword, date, or author) or marked for later reading.

Resource Databases. A set of databases containing activities and resources suitable for the classroom is available for biology, chemistry, physical science, earth science, and astronomy. Each science resource database includes lab descriptions, AV materials, field trips, and pamphlets. Each database record is fairly short, but longer, more descriptive files associated with some records can be retrieved by a single keystroke. Databases may be searched in a variety of ways, including scan by keyword and type of resource. Other databases include "roadshow," which describes speakers willing to visit classes; "whois," which contains detailed information on each of the MicroNet subscribers; and "proj.ideas," which lists science project ideas and appropriate resource materials or people. In addition to teachers, personnel from N.C. Wildlife, Sea Grant, and other agencies add items. Input into databases is possible from any account and is highly encouraged by the MicroNet staff.

Tutor/Testmaker. These two features make use of a special set of databases consisting only of multiple choice questions, grouped according to grade level and subject matter. The tutor portion randomly selects up to ten questions from a chosen topic (biology, chemistry, math, etc.) and subtopic and interactively quizzes the user. Two attempts are permitted for each question. When the test is completed, the grade is displayed and recorded into a file located in the student's directory. Immediate feedback (after each question) informs the person taking the quiz which answers were correct/incorrect. The tutor is primarily for the use of secondary school students.

The testmaker, available only for teachers both on the secondary and college levels, makes use of the same tutor questions for 'quick' off-line tests. The teacher may choose any number of questions which will be saved in a file, again located in his home directory. Printed out, this file is a properly formatted test with questions successively numbered. An answer key appears in the file, immediately following the test body. In order to keep the test database alive and growing, teachers are encouraged to input questions. Database size is limited only by disk space.

Activities

Conferences/Workshops. WCU MicroNet has hosted nine conferences and workshops on the Western Carolina University campus beginning with the first in March of 1982. Each teacher is encouraged to bring one or two outstanding students to participate in each on-campus activity: these students become tutors for other students and teachers back in their home schools. Conferences include three to four one-hour slots, each with three or more concurrent sessions. Smaller workshops, usually focusing on a single goal, have dealt with the Apple IIe computer as a result of an Apple Education Foundation/MicroNet grant which placed thirty complete Apple computer systems in fourteen western North Carolina schools. Conferences help keep teachers and students informed of new MicroNet features while workshops have dealt with topics such as computer interfacing and robotics. The network then provides a follow-up mechanism by which teachers share problems and successes associated with using what they have learned. On-line conferencing is used to determine what teachers want for in-service workshops and when these should be scheduled.

Communication. The network provides continuing support for teachers, since university faculty, teachers, and other professionals are on-line each week. Several students have established MicroPals which is conducted much like an electronic pen-pal network. On occasions, scientific data was collected, shared, and analyzed; photos, rocks, and plants were swapped. Friendships were formed and students gained a greater understanding of both the computer network and science. To paraphrase one of our teachers, through MicroNet, the computer is not a box, but a window--a window to people and resources.

Microlib: Access to Hunter Library. An account ('microlib') is monitored weekly by campus library personnel. Students and teachers at active schools can request copies of journal articles (not normally in a high school library) on a cost recovery basis. This is particularly useful for remote schools--both in the mountains and on the coast.

Data Collection and Analysis. Students and teachers at schools across the state have actively participated in data collection with scientists from the university and the North Carolina Wildlife Commission. Rain and streams were tested for changes in pH; hard mast (acorns, nuts as a wildlife food source) was counted; and anglers were surveyed on trout fishing and stream stocking practices. It is the intent of such projects to help teachers and their students do "real science" and analyze meaningful data under the direction of research scientists.

Quiz competitions. Beginning with "bioquiz," the MicroNet quiz competition has grown to include chemistry, physics, and earth science. Each quiz is hosted by a WCU faculty member with the exception of "physquiz" operated by Dr. Karen Johnston at North Carolina State University, Department of Physics. At the beginning of each week, a new question is posted; students read the question(s) and develop a solution or hypothesis; a student or teacher at each participating school sends the group response via electronic mail to the quiz moderator. The moderator replies (usually the same day) with comments designed to encourage and to stimulate further thinking. If two or more schools have equally good answers, the winner is based on speed of response (ie. the first correct answer receives the highest credit). At the end of each semester, the winners and honorable mentions are announced. Winning schools receive prizes and newspaper publicity for their efforts. Teachers frequently assign extra credit as an additional reward for participation.

File sharing. Many of the best inputs from our MicroNet subscribers involve relatively long file listings. For example, a lab exercise or a computer program listing might be several pages long. In order for others to read and use such lengthy documents, and to keep the telephone costs from skyrocketing, file sharing becomes necessary. MicroNet subscribers can share by uploading (sending files from their personal computers 'up' to the MicroNet computer) and downloading files (retrieving files from MicroNet 'down' to their personal computers).

Course Offerings.

Rationale. The original purpose of the MicroNet course was to allow secondary school teachers (and other non-traditional students who would not ordinarily be able to do so) to enroll in graduate level science courses. Mountainous terrain and sparse population decrease the liklihood that enough graduate students can be found who are willing to drive several hours one night a week to take a science course. As a result, teachers needing renewal credits have often signed up for more general education courses which can attract students from many disciplines. MicroNet not only makes taking courses more convenient--it permits offering of courses not otherwise possible.

Graduate courses delivered primarily by computer have met with great enthusiasm. A total of six graduate level science courses including Biology/Chemistry, General Ecology, Recombinant DNA, Chemistry for Teachers, Earth Science, and Stream Ecology have been offered, with enrollment increasing to 15 in the course offered this semester. Students (teachers or other graduate students) interested in the MicroNet course register and pay as they would for on-campus classes. Originally, continuing education credit was awarded; however, full resident credit for a graduate level science course is now assigned upon completion.

Course Description. Except for a few visits to campus, the graduate students take the course from home or school. An individual account is established for each registered student. Since MicroNet remains in operation 24 hours per day, students may login at their convenience. MicroNet courses differ from correspondence courses in that the instructor and other class participants can interact daily through the computer network.

Students communicate privately with one another or with the instructor by electronic mail; they participate in class discussions by reading and posting messages on the conference "bulletin board" created specifically for each course. Generally, the instructor enters a question or statement regarding the current subject matter as a starting point and, upon reading the initial message(s), each student may choose to respond to the instructor's or other students' notes. No limit is placed on the number of messages posted by each student.

Exams are distributed either through land mail or during campus meetings and are primarily multiple choice. Once a student has taken an exam on paper, he must access the system's exam module which interactively prompts with only the question numbers, making it necessary to have the test in-hand. Answers may be verified, or changed, after each set of ten has been input. Once all answers have been entered and verified, the module displays the score and the question numbers missed. The student is sent, through land mail, a test analysis which lists brief comments directing him to more information on each incorrect answer. Each exam may be taken a limited number of times as set by the instructor at the start of the semester. The instructor may choose to also assign essay questions to which students respond either by electronic or land mail.

Course Delivery Problems Encountered and Overcome. Earliest courses were conducted without initial class meetings prior to the final exam; however, we soon learned that computer interaction was much enhanced by bringing instructor and students together at the beginning of the course. On the average, three campus meetings are held per course. The first consists of a lecture, a hands-on introduction to MicroNet, and laboratory activities if appropriate. Intermediate campus meeting(s) may include lectures, labs, or field work. The final exam and course evaluation are usually administered during the last class meeting.

Teachers in early courses complained that they could not get access to computers at their schools at times when they could work on the course. We have solved this problem by purchasing fourteen Tandy Model 100 notebook computers which can be checked out for the semester. A telephone is the only additional equipment needed to use the system at any hour.

Instructor responsibilities. Since the courses are taught on-line, any class meetings are scheduled by the instructor. With the exception of the reduced number of class meetings and the extra analysis material for each exam, the responsibilities for the instructor are much the same as for an ordinary campus class. He selects and schedules all printed materials such as text books and reading supplements and makes them available to students on or before the first day of class or later via land mail. The instructor prepares exams well in advance of the test deadline so the tests may be distributed during campus meetings or through land mail. For each test he must also write an analysis--a brief, one line statement for each incorrect answer to direct the student to further information. The instructor enters this analysis, along with the exam key, into the computer. In conducting the course through the system, the instructor is responsible for beginning and monitoring discussions via the conferencing system and for responding to private correspondence via electronic mail.

Student responsibilities. Students taking a MicroNet course have somewhat different responsibilities than they would if they were enrolled in a campus class. Attendance at on-campus meetings, usually on Saturdays, is mandatory. Since there are few scheduled meetings, each student is required to access the MicroNet system weekly and interact with it either by inputting messages into the conference or by taking an exam. Logins are recorded by the system accounting and in some instances have been included in the grading scheme. Students must read and ask questions without the benefit of lectures on most of the course material. Since there is no set "class time," each student must determine a study schedule which provides adequate preparation for each examination. Some teachers (students) have started a course and have been unable to function in this environment; others appreciate the flexibility and have taken more than one MicroNet course. Of the 15 students currently enrolled, five have taken at least one MicroNet course in the past. In general, we think self-motivated individuals best adapt to our computer delivery of coursework.

Problems Still Facing MicroNet

Hesitation from Subscribers. With the increase in computer literacy among teachers, we have more requests for accounts than we can afford. Once on-line, however, teachers are reluctant to share their expertise even though MicroNet features require a minimal amount of data entry per user to maintain a steady flow of current and useful information. Although teachers comment on how useful they have found the databases and tutor/testmaker, they hesitate to input any of their original material, probably because they are pressed for time and doing so brings little reward. We are trying to find ways to encourage resource sharing, through summer stipends, some form of recognition, or as part of the "career ladder" professional growth plan required of some N.C. teachers.

Local Hardware/Software Difficulties. Most of the difficulties experienced with the main computer (Tandy 6000) seem to be inherent in the Tandy 6000 / Xenix operating system combination. The Tandy 6000 was designed for an office environment which normally includes only terminals, not modems. In effect, the machine cannot recognize a hangup (disconnect signal) and therefore cannot automatically log the user out. This occasionally results in accounts being left open and unguarded so that another person may enter the open account without a password. Also, at least one of the three ports connected to the WATS lines commonly becomes inoperable during the nights and weekends and remains unusable until the system operator takes action to correct the problem.

Expansion in any direction (additional accounts, simultaneous graduate courses, other disciplines) will have to wait until funds are secured for a larger computer. The need for a larger and faster system has been made obvious by the extremely slow reaction times, the lack of sufficient disk space, and the limited number of ports. Expansion into other disciplines will require additional ports for campus-wide terminals and is also dependent upon campus networking which is now underway.

Uploading/Downloading due to Differences in School Configurations. Very few schools have identical combinations of computer, modem, and communication package. This variety makes it difficult at best for us to help each school with uploading and downloading of files since these processes are software-dependent. As a result, most users type all messages directly on-line, which increases our telephone costs.

Computer Hackers. The WCU MicroNet system was designed specifically not to attract the attention of computer hackers. MicroNet is an open system which allows any user to reach any level desired by only two keystrokes. Since Xenix is inherently difficult to secure, we chose to leave the system open on the theory that if it is readily available to anyone, penetrating the system will no longer be a challenge. We have occasionally experienced a few unauthorized uses such as prank mail messages and obscene data entries. In most cases, a phone call to the student's teacher and a change of password was a sufficient remedy. Isolating the network on a stand-alone system removes the temptation of "hacking" into student grades, payroll, and other information which were on the University mainframe.

Funding. Funding remains a number one concern, for networking is expensive. Start-up costs for the stand-alone system included approximately $30,000 for hardware and software purchases, $16,000 per year for a full-time system operator, (NSF Grant - #DPE-8319910); hardware maintenance, office personnel (part-time director, research assistant, graduate student, and at least one student worker); office space, and supplies, postage (WCU Chancellor's office) and telephone charges (WCU and Title II Grant). Selected secondary school teachers participating in summer development projects in both 1985 and 1986 were paid stipends of $500 per week (NSF) to create new projects or programs, improve old ideas, research materials necessary for implementation of old and new projects, and begin putting the new ideas into use. Both summers were extremely successful: teachers paid for one week of summer work were enthusiastic and continued to contribute to the system throughout the academic year.

Our faculty is traditionally dedicated to helping pre-college teachers and continues to do so. The WCU faculty members who are actively involved with MicroNet currently receive no automatic salary increase for their participation during the academic year; they were paid a summer salary (NSF grant) for working with teachers on stipend during the 1985 and 1986 summer sessions. Some reward system beyond the recognition of their service on faculty evaluations may be necessary to sustain faculty effort over the years; however, at present we continue to recruit volunteers both from our faculty and from museums and other support institutions for which MicroNet is a way to communicate with teachers.

One of the largest continuing operational costs which grows in proportion to network use is the telephone charges. Because our schools are scattered and almost all must call long distance, our annual phone bill for the 1985-86 school year amounted to approximately $18,000. As justification for payment of such formidable telephone bills by the University and Title II grows increasingly difficult, we must begin to locate alternate long-term funding. Suggestions under consideration include a set subscription fee charged to each active school, charges proportional to the amount of on-line use billed to each school, discontinuation of the three WATS lines (change to local telephone lines) so each school pays its own bills, and tax deductible donations made by companies and corporations within each school's region. We are also examining the possibility of funding by the legislature as a special project. We have been reluctant to bill schools for long distance charges since small schools with the most to gain from networking are least able to pay and may be forced to leave the system.

Future Plans

We continue to attack the problems stated in order to expand the network to additional schools and disciplines. Expansion is controlled at present due to limitations of hardware and telephone expenses. We are actively seeking acquisition of a larger and faster main computer which would be used as a front-end to the existing Tandy 6000; programs were designed to make transition to a larger system relatively painless. With the proper equipment, expansion into other disciplines will be simple since the software designed and written at MicroNet is not confined to the sciences. It is generic in the sense that the conferencing, database, and tutor programs are tools which may be put to use in a variety of different settings. English and health sciences are among the disciplines which have expressed interest in becoming involved with MicroNet.

A MicroNet simulation disk (for the Apple IIe) is being produced to familiarize new subscribers with the system and to demonstrate the network to a large group of observers simultaneously. The ideal simulation will give the user the impression of actually being on-line, in effect training him to use MicroNet without the high telephone costs that have accrued in the past.

If funding difficulties can be surmounted, the MicroNet networking model provides an effective vehicle for making resources and expertise accessible to pre-college classrooms. We anticipate addition of non-credit modules through which teachers can update their knowledge in specific content areas in response to personal interests and professional needs. The delivery of coursework promises new opportunities for life-long learning--not just for teachers--but for anyone whose responsibilities, interests, or physical handicaps prohibit enrollment in traditional college programs.

TEACHING A 19th CENTURY MODE OF THINKING THROUGH A 20th CENTURY MACHINE

C. A. Bowers

University of Oregon

The use of microcomputers in the classroom involves a profound irony that has escaped the attention of those who write the educational software programs and the monographs that attempt to explain the uses and limitations of the new technology. The irony is that the software programs for these 20th century machines reinforces a 19th century view of knowledge and the individual. A probable explanation for this oversight, which can hardly be viewed as insignificant, is that the people who produce the programs and the literature on the educational uses of microcomputers have unquestionably accepted an interacting set of myths about the nature of technology, the character of knowledge, and the nature of the individual. As these myths underlie the taken for granted assumptions that guide most educational thought and practice, it is not being suggested here that the advocates of educational computing are more controlled by these cultural myths than others. The basic problem, aside from the personal one facing program writers who do not recognize how their own thought process is culturally conditioned, is that the unique power and versatility of the microcomputer will further bind students to a 19th century pattern of thinking and set of assumptions that have been discredited in many areas of the academic world.

In order to put the problem in clearer focus it is necessary to recognize that there are three general areas of expertise regarding the development and use of microcomputers in education. These include the development of the computer languages that run the software programs, the writing of software programs that perform specific educational tasks, and what can, for lack of a better phrase, be referred to as the non-technical aspects of educational computing. In terms of this tripartite division of the territory, the latter area includes understanding how the educational uses of microcomputers interacts with and influences the cultural patterns that constitute the environment of the classroom. These include the patterns of social interaction, the legitimation of what constitutes knowledge, the political ideology reinforced by the content of the educational experience, and how the technologies used in the classroom mediate and transform the cultural transmission process. Although this third area of expertise, which really has to do with the cultural, educational, and political implications of microcomputers, is the most complex and difficult one of the three it is, interestingly enough, nearly totally ignored in the professional educational programs dealing with computers. But that is the subject of another article. Our purpose here is to stress the point that the following analysis of how current educational software reinforces a 19th Century mode of thinking is only part of an area of inquiry that should be an integral part of the teacher and software writer's "computer literacy."

The argument that the use of the microcomputer shapes the student's way of thinking, as opposed simply to facilitating it, is contingent upon settling the issue of whether technology is a neutral tool. The familiar aphorism "garbage in, garbage out" that one continually hears in discussions about the influence of microcomputers succinctly represents the orthodoxy held by most educators that microcomputers are a neutral technology. The microcomputer simply facilitates and expands our symbol manipulation capacity, and it does it with a degree of speed and reliability that we cannot match. As we program these machines, we control them in much the same way we control how the pencil is to be used (an analogy that educators like to make). This is a totally erroneous view of microcomputers and, for that matter, of the neutral nature of pencils; in explaining why this view does not hold up we can begin to see how the educational uses of microcomputers reinforces a particular pattern of thought--one that has important affinities with the Newtonian world view that is now undergoing fundamental revisions.

The easiest way to understand the non-neutrality of a technology is to follow Don Ihde's suggestion that we consider how experience is mediated by the technologies we use. Simply put, he asks us to consider the technology (i.e., microcomputers) as part of the field of our experience and to pay close attention to what aspects of experience are selected, amplified, and reduced through interaction with various forms of technology (Ihde, 1979). How the use of a particular technology mediates and transforms the nature of our experience can be understood, to start with the simplest example, by looking at what aspects of experience are amplified and reduced when we use a pencil. Use of a pencil amplifies our ability to express our thoughts in written form, and because of the characteristics of this technology we have the time to reformulate our thoughts in the process of writing them down. By opening the avenue of written expression the pencil amplifies a whole series of characteristics that have social, cultural and political consequences: a privatized form of communication, a decontextualized form of thought, creation of a text that takes on an independent existence thus allowing for critical analysis, and communication with an anonymous public (Ong, 1982). At the same time, the use of the pencil reduces (selects out) those aspects of experience connected with sensory awareness and tacit forms of understanding. In the larger scheme of things, the pencil amplifies those aspects of experience that foster individualism, analytic thought, and the reification of the word; but it reduces the human capacities that are expressed in oral traditions: context specific sources of meaning, the full use of all the senses, and the spoken word in an ongoing process of negotiating meanings with others.

To take another example, we can ask what does the use of the telephone amplify and what does it reduce. It's a powerful technology for communicating our voice over great distances, as it reduces other aspects of the communication process it sharpens our tendency to listen carefully. But it reduces our ability to use context, body language (including facial expression) as part of the message system. In learning to think of how different technologies--automobile, fork, book, calculator, flute, etc.--amplify certain aspects of experience, while reducing others, it becomes less strange to ask what a microcomputer, given the current state of software, amplifies and reduces. But in order to understand the educational significance of this line of questioning we need to put in focus a more complex view of experience--one that takes account of the cultural aspects. Thus before we can examine what the use of microcomputers amplifies and reduces we need to situate this technology in terms of how culture is transmitted and experienced in the classroom. This will enable us to see what is being amplified and reduced, and how this selection process (which involves the microcomputer acting on the student) reinforces a 19th century mode of thought.

Briefly, all aspects of human experience are influenced and sustained by culture: technologies, customs and norms, political and economic institutions, and so forth. But the aspect of culture most pertinent to understanding the amplification--reduction function of microcomputers is the symbolic. Although not all aspects of culture have a symbolic dimension the aspects most important to our discussion have to do with the language systems--verbal and nonverbal that provide the information codes for thinking and acting (Goodenough, 1981). In learning to speak we acquire the conceptual categories and assumptions of language community, and through the ongoing conversations with others the culturally derived patterns that govern interactions, purposes, and achievement are continually reinforced and modified in minor ways. The key point here is that language, as Claus Mueller (1973) put it, "is a repository of cultural traditions;" it provides the conceptual basis for our interpretations, reflective thought, imagination, and thus choices and behaviors. There are three implications of this statement that need to be made explicit: first, knowledge is an interpretation based on the conceptual categories we unconsciously acquire when we become members of our language community; second, most of our knowledge of how to think and act is based on tacit understandings where performance and context are more critical than our ability to make explicit the knowledge we possess (Dreyfus and Dreyfus, 1986); third, language provides the symbolic framework that influences human actions--including the creation of institutions and material objects. The implication of this last point is that everything that is humanly created has a history, and that it embodies in the present, elements of the symbolic order out of which it originated.

By emphasizing the symbolic aspects of culture we can more easily view the classroom as a complex language environment, and thus see more clearly that what goes on in this environment is the transmission of the conceptual patterns and information codes of the dominant culture. The transmission is exceedingly complex and includes talk, reading, use of space and time as an information code, regulation of patterns of social interaction, and use of body language. Part of the transmission process is explicit (the lesson, textbooks, the point the teacher wants to make, the issues that are argued, and so forth) but much of the culture is learned and reinforced at the tacit level where neither teacher or students are fully aware of the cultural patterns and assumptions that are being learned.

By considering the classroom in terms of a complex language environment we have avoided the mythic starting point that the classroom is made up of autonomous, self-directing and rational individuals, some of whom find their way over to the microcomputer for the purpose of using the data base as a source of objective information or the word processor to express their individual thoughts. In the classroom both teacher and students operate within the limits of language:--imagination, reflection, intuition and interpretations are constrained by the conceptual categories of language. But these limits are also the basis for new possibilities.

Having situated the student and the microcomputer within the context of the complex language environment that characterizes a classroom we can now turn to the question of how does the microcomputer mediate and shape what goes on in the cultural transmission process. We could discuss the amplification-reduction characteristics of word processing, but the use of a data base like NewsWorks will serve as a better example of how knowledge is structured through the characteristics of the technology and, in turn, how the computer technology reinforces a particular pattern of thinking.

What the software amplifies is clearly stated by the creators of NewsWorks:

> By using data bases in the classroom, students develop critical thinking skills and improve their ability to use information retrieval systems. Using data base samples encourage students to:
>
> -compare and construct data
> -determine cause and effect relationships over time
> -make inferences from data
> -form and test hypotheses made from data analysis
> -predict historical and economic trends from data
> -develop inductive and deductive logic by using data
> -improve research skills" (1985).

This is indeed an impressive set of claims, and because the assumptions about the nature of knowledge and how we think that are embedded in this statement are so widely shared by software program writers and teachers, few are likely to see how the use of this technology will shape the student's way of thinking. What is amplified through the use of the data base is the representation of knowledge as discrete bits of information (facts) in a manner that leads to objective conclusions (truth). The data base also amplifies the ability to collect, store, and retrieve data that represents what can be observed and measured. The data base presents a de-contextualized and thus abstracted representation of what the words on the monitor signify. Thus the technology further strengthens a long standing cultural tradition of accepting the printed word (and measurable data) as an accurate presentation of reality.

We should now turn to the aspects of experience that do not get transmitted as the students sit in front of the microcomputer manipulating the data files. As an algorithmic system, it is impossible to program forms of knowledge that cannot be made explicit and organized into discrete components, and where the operational rules cannot be formally represented. Thus the machine in front of the student cuts out of the communication process (the reduction phenomena) tacit-heuristic forms of knowledge that underlie common sense experience, the awareness that knowledge is an interpretation that is influenced by the conceptual categories embedded in the language of the person who discovered or established the knowledge as fact, the recognition that language and thus the foundations of knowledge itself are metaphorical, and finally that the data have a history. The binary logic that so strongly amplifies the sense of objective facts and data based thinking serves, at the same time, to reduce the importance of meaning, ambiguity, and perspective. The sense of history and the cultural relativism of the student's interpretative framework are also put out of focus.

In effect, the use of the computer data base mediates the cultural transmission processes that go on between the student and the social world. This can, of course, be said of all technologies, for example, eyeglasses mediates our relationship to the environment in a manner that amplifies our ability to see more clearly but does not strengthen our acuity in areas of taste, smell, and sound. But what is important for educators to consider is the pattern of thought, and image of individualism, that is reinforced by what the microcomputer amplifies and reduces in the cultural transmission process. The claims made in behalf of NewsWorks, and which are also used to justify other software programs, involve assumptions about the nature of knowledge that were established as part of the positivistic tradition of thinking that had its roots in the 19th century. At that time Auguste Comte and Emile Durkheim (two French sociologists) were helping to lay the conceptual basis for a positive form of knowledge. The scientific method provided them a powerful model for how to think about the social world; thus the concern with reducing experience to the observable and measurable, and in examining the variable relationships. The emphasis on objective knowledge, as well as the tendency to interpret cultural differences within an evolutionary framework led to a view of knowledge that is essentially identical to position contained in the quotation from NewsWorks.

By amplifying a 19th century view of knowledge the data base reduces (selects out) the elements of understanding that are associated with developments in the 20th century. These include an understanding that what is experienced as "real" is socially constructed (Berger and Luckmann, 1967), and that our interpretations are influenced by the conceptual categories embedded in our natural language. This view of language forces a recognition of the relativism of the interpretative framework used to explain what the "data" means. In recent years there has also been a growing recognition that the problem of meaning has been put out of focus by the emphasis on an objective world. The problem of meaning (i.e., how the individual interprets and gives meaning to daily life) is fundamental to whether students can make generalizations about the peoples of other cultures, even when the generalizations are based on "objective data." Generalizations about other cultural groups ignore the taken for granted interpretative framework of the student, the data collector (often hidden from consideration), as well as the interpretative framework of the people who are the objects of the generalizations. Lastly, there have been important developments in understanding the difference between explicit and implicit forms of knowledge. Recent advances in anthropology, philosophy and the sociology of knowledge have provided us a way of recognizing the knowledge we have difficulty making explicit because of its taken for granted nature (Douglas, 1979). This is knowledge that we learn from others in contexts, and is basic to successful cultural performances (when it's appropriate to tell a joke, how to adjust our sense of space in accordance with changes in context, etc.). It also provides a secure taken for granted world of shared patterns that allows us to make explicit and reflect upon specific areas of cultural activity.

The 19th century view of knowledge that is amplified in the NewsWork data base, as well as in other educational software programs that provide factual information for reasoning, creates a number of educational problems that are related to the mythic view of objective knowledge. One problem relates to the view of individualism that is reinforced by the 19th century convention that separated the knower from the known. The view of thinking as information processing seems to fit nicely with the liberal view of the individual as an autonomous, self-directing being. To insure that self-direction is based on a reflective process it is essential, according to this view, that the individual be given access to all the facts. Thus the capacity of the microcomputer to store, manipulate and retrieve data makes it the ideal educational tool for facilitating the individual's rational capacity for self-direction. The only problem with this view of the individual is that it is based on a number of misconceptions that go far back into Western thought. For our purposes, we can simply point out that if the individual speaks a language, she/he will think within the conceptual categories and assumptions that underlie the culture's world view and thus cannot be considered as autonomous or entirely self directing. But more important than these problems, which have to do with reinforcing a mythic view of the individual, is the civic issue of how to educate in a manner that strengthens students' ability to recognize themselves as members of a larger social group--membership provides the cultural resources that will be individualized as part of their own personal growth; it also carries with it definite shared responsibilities for strengthening the cultural resource base. Learning about self as part of a "community of memory," to use Robert Bellah's (1985) phrase, is undermined by the view of the autonomous individual who only needs objective information in order to be rational.

Another major educational problem raised by the use of the microcomputer in the classroom has to do with the role of the teacher. The data base, for example, far exceeds the teacher's capacity to provide relevant and accurate information. The use of other software programs, which rely upon a degree of expertise that few teachers can match, makes the teacher's role even more ambiguous. As the use of computer-assisted instruction becomes more widespread, the question of what the teacher can contribute to the educational process, beyond providing guidance in the use of the technology, becomes a more paramount concern. If we stay with the objective form of knowledge that fits a computer-based system of instruction the teacher's role will be that of a monitor of student behavior and a technical advisor. But if we take seriously a 20th century view of knowledge--that facts represent the objectification of somebody's interpretation, that interpretations are influenced by the conceptual guidance system of a culture, that objective knowledge is about a world of events and objects that become more fully understood as we trace their historical development, that language and thought are metaphorical in nature, that explicit-calculating forms of knowledge are quite different from the tacit-heuristic forms of understanding the problem solving, and that the structure of knowledge and individual understanding are shared and filtered by what individuals take for granted--it then becomes possible to recognize the unique contribution that only the teacher can make to the educational process.

The teacher cannot match the machine when it comes to the tireless reproduction of factual information, or the knowledge base that can be made available through the microcomputer. But only the teacher can amplify those aspects of the cultural transmission process that are reduced by the selective characteristics of the microcomputer. This involves restoring to teachers responsibility for recognizing when to intervene in the educational process by clarifying the conceptual problems that arise from metaphorical thinking, by pointing out that the objective knowledge reflects somebody's interpretative framework and that this framework must be understood in terms of the underlying cultural assumptions, by making explicit the student's taken for granted assumptions and how these assumptions influence understanding, by guiding the student to think in terms of historical continuities and transformations, and finally, by urging students to check the more abstract and context free forms of knowledge (data, generalizations, etc.) against their own common sense forms of understanding. This is the aspect of the cultural transmission process that cannot be programmed because it involves paying attention to what the student and program writer are imposing upon the data, the context within which learning occurs, and making the conceptual connections between what is being learned and other parts of the curriculum.

In effect, a 20th century view of knowledge involves using the microcomputer as a powerful and legitimate tool of the teacher and students. But it means subordinating the machine to the complexity of the human experience rather than amplifying only those aspects of experience that fit the logic of the machine.

References:

Bellah, Robert N., Madsen, Richard, Sullivan, William, Swidler, Ann, & Tipton, Steven. (1985). *Habits of the heart: Individualism and Commitment in American Life*. *Berkeley: University of California Press*. p. 153.

Berger, Peter & Luckmann, Thomas. (1967). *The social construction of reality*. *New York: Anchor Books.* Especially the chapter, "Society as Objective Reality."

Mueller, Claus. (1973). *The politics of communication*. London: *Oxford University Press* p. 15.

Douglas, Mary. (1975). *Implicit meanings*. London: *Routledge & Kegan, Paul.* Rorty, Richard. (1979). *Philosophy and the mirror of nature*. New Jersey: *Princeton University Press*.

Dreyfus, Hubert & Dreyfus, Stuart. (1986). *Mind over machine: The power of human intuition and expertise in the era of the computer*. New York: The Free Press.

Goodenough, Ward. (1981). *Culture, language, and society*. Menlo Park, CA.: *Benjamin/Cummings Publishing Co.* p. 66.

Ihde, Don. (1979). *Technics and praxis*. Boston: D. Reidel *Publishing Co.* pp. 56-57.

NewsWorks: Teacher's guide. (1985). New York: *Newsweek Publishing Co.* p. 1.

Ong, Walter. (1982). Orality and literacy: *The technology of the word*. New York: *Methuen.* pp. 78-108.

COMPUTERS AND BASIC ECONOMICS: DREXEL'S EXPERIENCE

Edward Clifford Koziara
Chiou-Shuang Yan

Professors of Economics
Department of Economics
Drexel University

Abstract

This paper describes and evaluates the adoption and use of personal computers at Drexel University, especially by the Economics Department. The Drexel experience may be of value to others contemplating computer aided teaching of economics to a student body fully equipped with personal computers.

Introduction

The president of Drexel University decided in October, 1982 to have all freshman acquire a personal computer starting in the Fall of 1983. Due to the short lead time, substantial university resources were mobilized to implement the decision. Immediately, a microcomputer selection committee, composed of faculty members, was created to define the needs of the university, to establish the specifications of a computer fulfilling those needs, and to identify manufacturers who could supply the computers. The final selection of Apple's Macintosh was then made by the administration.

Before the Mac computer usage was common in natural sciences and engineering; and to a lesser extent in the College of Business and Administration where the Department of Economics is located. Simulations were used in the basic marketing and management courses and statistical computations were done on mainframes. The Economics Department, however, had never used computers in teaching. The department functioned primarily as a service department. It provided introductory economics courses for all university students and had few majors.

Training and Support of Faculty

To prepare faculty for a successful microcomputer program, Drexel embarked on a massive faculty training and development effort supported by a $2.8 million grant from the Pew Memorial Trust. The specific measures taken are summarized below:

1. Training on Apple's Lisa: A number of Lisas, the Lisa was the predecessor of the Macintosh, were made available before the arrival of the Macintosh for faculty training. Faculty members were released from some teaching duties to be trained on word processing, drawing, painting, charting and the spreadsheet. Although training was not mandatory, it was encouraged. The faculty was initially enthusiastic and participation in training was almost universal at the business college.
2. Software Development: Faculty members were given released time to develop teaching software. A member of the Economics Department developed homework problems for basic micro and macro courses. Another member received support to develop classroom examples illustrating various economic concepts.
3. Developers' Sessions: University-wide developers' sessions have been conducted since 1982. Usually a session is given each week at noon in which participants bring their lunch to encourage maximum participation. In each session, a work in progress is presented for comment and evaluation by the participants. Members of the Economics Department participated actively in these sessions as presenters and commentators.
4. Seminars: Seminars were organized by a member of the Economics Department for the faculty members and graduate students in the College of Business and Administration. Knowledge and information on new software either developed by others or by members of the business faculty were exchanged.
5. Boot: A weekly campus newsletter has been published since 1982. Boot provides information on computer activities and developments such as bugs in software, cures for bugs, software reviews, the availability of new materials, brief reviews of books and speeches relating to computer aided instruction.
6. Software Review Center: Established before the arrival of Macintosh, the center was expanded to meet the new needs. The center provides references, collecting and classifying existing software, manuals, periodicals and books.
7. Instruction Support Group:

The group was established in 1982 to assist faculty in software design. The group offers programming support and offers professional help in evaluating the technical capabilities of existing software and their compatibility with different hardware.

8. Policy on Software Royalty: To encourage faculty development of software, the University proposed to split royalties, 40% to 60%, with the faculty. The 40% received by the University will be allocated partly to the University's general fund and partly to fund other software development. Arrangements have been made to market software through Kinko.

Student Support

Students were able to purchase hardware and software at reasonable prices negotiated by the University and Apple. Additionally, computer clusters and printers were made available at various campus sites. Sites were staffed by student consultants to assist users. Students could also avail themselves of special mini-courses in software applications offered by the Instructional Support Group. For example students taking economics might sit in on a one or two session course on the spreadsheet. There is also a MacUser club which is quite active on campus.

Early Experience in Using the Macintosh in Microeconomics

Four introductory economics courses are offered at Drexel: microeconomics and macroeconomics for engineering and science majors carrying three credit hours each; microeconomics and macroeconomics for non-engineering and non-science majors carrying four credit hours each. Teaching with computers was initiated when the first group of non-engineering and non-science majors, composed mainly of business students with the Macintosh started to take microeconomics in the spring of 1984. The other three courses were taught in the standard way. In the computer course it was decided to use the Macintosh in the classroom as well as assigning ten homework problem sets to be done on the Macintosh. The decision was made by a group of faculty interested in the project rather than the department as a whole. Multiplan spreadsheets were used exclusively for both classroom presentation and homework assignments. Since there were only a few large lecture rooms equipped with the Macintosh and projection facilities, the four small weekly sessions were consolidated into one large lecture and three small sessions. Although most faculty members in the department trained the previous summer, only a few were proficient enough or had a desire to handle the Macintosh in the classroom. Thus, only those in charge of the large lecture were required to apply the Macintosh and were responsible for the homework and 25% of the grade. There was virtually no coordination between the lecturer and instructors of small sessions.

Problems

In the initial application of the Macintosh, there were problems due to inexperience, a lack of planning and the overuse of the Macintosh simply because of available facilities. Major problems encountered were as follows:

1. Facilities: In a large lecture room seating up to 400 students a Macintosh was hooked up to a projector which displays the Macintosh screen on a 12' x 12' screen in lieu of a blackboard. Because of poor projection lights must be off when using the screen preventing students from taking notes. Poor visibility and the large number of students generated noise from the students when lights were off for more than five minutes. Later, this difficulty was partially remedied by using the screen for only a few minutes at a time. Instead of using the screen overzealously, it was used only when needed.

2. Student panic: Most of the students were freshman still feeling their way in college. They were upset about having to learn economics, operate the Macintosh and use the spreadsheet at the same time. They felt, and were, guinea pigs; but, so were the instructors. Adding to the difficulty were the bugs in the version of the Multiplan spreadsheet used at that time. Student complaints subsided after graduate students were made available for tutoring, Multiplan hands on sessions were offered, and efforts were made by lecturers to be particularly sensitive to the needs of students venturing where none of their predecessors had been before.

3. Mixing of students with and without Macintosh: Only freshman were allowed to purchase the Macintosh. The upper classmen taking the course were not separated from the freshman. Yet, the upperclassmen were required to do the same work the freshman were doing. This required access to a computer. Arrangements were made to have the upper classmen use university Macintoshes at the various campus clusters. Still, the upper classmen had misgivings because they had not taken the course earlier in their Drexel career they felt they were penalized in having to do extra computer work. At the same time some were irate that they were not allowed to purchase the Macintosh and software as the entering students. Provision for purchase by upperclassmen was not made until the next year.

4. Copying homework by the students: Some students copied hoemwork from others. To reduce this practice, no Xeroxed copies of homework were accepted and the lecture exam included questions on using Multiplan. This problem was later dealt with by assigning individual parameters based upon student numbers.

5. Logistics Problems: Homework problems were handed out, collected and returned each week in the lecture session. In addition, paper copies of the (Multiplan templates) examples presented in class were distributed to the students. The sheer number of students and

volume of paper involved made the process rather cumbersome and time consuming. This procedure was changed in the fall of 1984 by handing out homework problems in a complete package at the start of the term; problems were collected and returned by the small session instructors. Instead of handing out paper copies of the class examples, disks containing all class examples were made available at the computer center for students to copy or use in the computer center.

There were also some positive aspects of using Macintosh. Economic analysis lends itself naturally to spread sheets. Capability of spreadsheets in handling changes in parameters greatly facilitate impact analysis. Student interest and concentration increased as the electronic blackboard flashed the result of a change in an economic variable. The homework problems allowed student hands on experience in analyzing more complicated and realistic economic problems which enhances their understanding of economic theory. Many students were excited doing homework on the Macintosh. Previously, there were no homework assignments and students were graded only on exams. The transition from the traditional way of teaching economics also went on smoothly without friction among faculty members. This was due to the fact that faculty members were not compelled to use the Macintosh and the lecturers and section instructors did not spend extra time on coordination because they each conducted their teaching independently. This lack of coordination, however, was not always palatable to the student.

Using the Macintosh in Macroeconomics

Lack of graduate assistants prevented the use of the Macintosh in the summer of 1984. In the fall of 1984, microeconomics for non-engineering and non-natural science majors was taught with the same format used in the spring of 1984. In addition, macroeconomics for non-engineering and non-natural science majors came aboard with computer. There was no major problem encountered in microeconomics simply due to prior experience. Although lessons learned before were also applied to macroeconomics, some unexpected problems arose. Extremely large class size in the lecture, a greater number of non-computer owners enrolled in the course, the failure to recruit the needed graduate assistants at the outset, and an over-ambitious homework design invited strong student complaints in the first week. Fortunately, graduate assistants were made available in the second week and non-computer owners were immediately given access to the Macintoshes in the computer clusters. To quell student fear, the first homework was graded on effort. By the second week student sentiment reversed. Numerous students said they were surprised how easily they could pick up Multiplan and thought the course a marvelous experience.

The 1985 Experience

In the winter and spring of 1985, computerized lectures were given for both micro and macroeconomics courses. However, lecture rooms with computer and projection facilities were not available. The lecturers were forced by necessity to see how they could conduct teaching with disks and computer facilities outside the classroom. By this time, students were better prepared and understood references to the Macintosh. We all started to understand the same language. Materials identical to classroom presentations, homework tips and a sample quiz, all on disks were made available to students. To enhance student interest, a copy of a data base disk was also made available in which economic data from 1947 to the present are contained.

In the fall of 1985, basic microeconomics, with Mac, for engineering and science majors was started. In addition to the ten problem sets designed for other students, three additional ones were developed specifically for engineering students to help them prepare for their professional certification. Unlike business majors who took the first economics course in the freshman year, engineering and science majors did not take economics until their junior year. They were generally more mature and familiar with computers. Thus no difficulty was experienced in adjusting to computer aided teaching.

At this time a member of the department received support to develop ten self study units for microeconomics. In each unit, a topic is reviewed through a series of interactive questions. Through a random number generator, students get different problems each time they use a self study unit. Students have their answers corrected instantaneously on the screen, are drilled on the concept with which they have difficulties, and can repeat an exercise until they feel comfortable. The computer also records the performance of students. These self study units are designed to replace the present homework problems turned in on printouts. They will automatically record the homework done, how well it is done and meanwhile correct mistakes and tutor students. The need for graduate assistants in recording and correcting homework is thus eliminated as well as the logistic problem of paper handling. The program is still underway.

Student Evaluation of the Economics Program

A survey of students taking the introductory economics courses was conducted in the fall of 1985. Of the approximately 600 students taking introductory economics courses, 385 returned the questionaires. Students were asked to evaluate the learning effectiveness of various means available, the ranking by students is summarized below in Table 1. Computerized homework, although ranked only 6th among first choices, was fourth in weighted average.

Table 1: Effectiveness of Various Teaching Tools

Weighted Ave. #		1st Choice	1st&2d Choice	1st,2d&3d Choice
7.68	Text	29.6%	52.7%	70.6%
4.24	Study Guide	3.1%	11.4%	18.4%
3.89	Old Exams	5.5%	11.2%	20.0%
2.37	Disks	1.0%	2.3%	5.2%
6.82	Lecture	20.3%	38.2%	52.2%
6.48	Small Class	21.8%	35.8%	48.8%
6.08	Homework	4.4%	17.4%	31.4%
5.28	Work/Others	3.6%	11.9%	23.9%
3.37	T. Asst.	5.7%	10.4%	15.6%
0.62	Other	2.3%	4.4%	4.9%

#If all students picked a category first, the weighted average would be 10, if they all picked it last, the weighted average would be 1.

Table 2: Location of Computer Use

Site of use	% of Respondents
Home only	12.99%
Home & small classes	5.19
Home & lecture	4.42
Home, lecture, small classes	9.35
Small classes	5.71
Small classes & lecture	.26
Lectures	4.16
Do not use in lecture-poor projection	9.35
Site Comments..................	51.43%

Students were asked to write an answer to the question "Where should the computer be used?" The comments of the 51% responding to this question are summarized in Table 2. The largest number of students preferred to use the computer only at home while nine percent of the respondents specifically objected to the use of computer in a large lecture room because of poor projection. Over nine percent of the respondents, however, felt that the computer should be used whenever needed and useful, implying the usefullness of the computer.

Although 7% of the respondents commented negatively on the homework, they are more than offset in favorable comments by an almost 3 to 1 ratio. To further improve handling of homework, a new homework text was adopted in the winter of 1986. The text contains 20 problem sets with fully detailed examples of format and calculation results. The variable parameters in each problem set may be changed within specified ranges, allowing instructors to change and assign different problems to different students. The text reduces the student panic at the beginning of a term, lessens the logistic problems mentioned above, and partially prevents students from copying homework.

Without specific solicitation, where students were asked to comment freely on the computer and economics, one out of five students specifically stated that the computer had been useful in understanding economics. Some called for use of other software in addition to the spreadsheet, they especially called for employment of graphics to understand various cost concepts. A member of the department has been working on software for graphic analysis. Units on determinants of demand and supply, elasticity, marginal utility and pure competition have been completed. Work is underway on monopoly, and production function.

1986: Developments

In 1986 the 20 problem sets for both micro and macroeconomics were packaged in a workbook, Solving Economic Problems with Spreadsheets, which further minimized the paper shuffling that occurred earlier. A second edition appeared at the end of the year.

Students in the general macroeconomic course were required to do a paper on International Trade. Students were offered an opportunity to take a one hour course in the operation of the Dow-Jones News Retrieval System and were given free access to the service. Students were told currency of information, substance and form would be equally weighted in determining the grade. Thus, information retrieval and word processing abilities were added to spreadsheet skills.

Additionally, a project, worked on by a department member, along with members of other departments, got to the funding stage. The project is a simulation that would be used by all business students in various courses throughout their stay at Drexel. The simulation would use real world data, be played either in real or simulated time, and would provide records of student performance, and a rich student and faculty research base.

1986: Computers, Economics and Test Performance

Although various explanations may be offered; there was no study of student test performance before and after the introduction of the computer. However we are now attempting to find out how well students perform on examinations if they are assigned computerized homework, have disks with class examples available, and are urged to master the spreadsheet applications. Our studies are at an early stage and we offer the findings of testing a group of students in the spring and summer quarters of 1986 (See Table 3.) The students were taking B412, macroeconomics, and they were mainly engineering and science students, who generally were exposed to the computer in their earlier economics course. The spring group was the computer using group of students and the summer group was not assigned homework, nor were the disks mentioned in the course outline.

To prevent copying students in lecture are given four different, 30 question, multiple choice examination. Only one lecture exam per term is given, the other exams are given by the small section leader. The examination format remained basically the same for the two quarters observed, with the exception that parameters were changed and question shuffled.

Table 3 gives the average grade for each of the four exams, and the number of students taking the exam. On each examination the students in the non-computerized group did better than those in the computerized group.

One reason for the result might be that the examination format was out and students could learn by doing old exams. This is a possibility, however we do not think this to be the case, since a similar format had been used for a number of terms, and all students were informed precisely on the topics that would be covered, the nature of the questions, and the number of questions on various topics. Could it be, that despite the analytical and mechanical nature of the test, that the computerized work was not conducive to better test performance?

Table 3: Computers Use v. Non Computer Use and Test Performance

Engineers	Spring 86	30 Point Maximum Score			
	Test I	Test II	Test III	Test IV	Total
average	23	22	25	22	22.6
students	55	58	52	50	215
Engineers	Summer 86				
	Test I	Test II	Test III	Test IV	Total
average	25	23	26	27	25.26
students	23	33	24	36	116

1987

Early 1987 saw the printing of the second edition of the workbook. Additionally a set of parameters, keyed to student numbers was prepared for all 10 macroeconomic problems. The Dow-Jones News Retrieval experiment continued. The one change being that now students were given five dollars of free time if they took the course, rather than given unlimited free time. The continual access business simulation game caused the College of Business to rethink its entire curriculum, an example of an impending technological change causing possible far reaching changes prior to funding or being put into place. This rethinking was partially due to a request from the funding group and partially due to faculty concerns.

The Future

There is a certain maturation that has occurred among administrators, faculty and students. The early excitement is no longer present. This lack of euphoria is a blessing and a curse. The Macintosh has become part of the furniture. Some of the spontaneity of the early years has gone, the computer is in place. Faculty are not provided with the newest machines and software. They are expected to make do as best they can. This can lead to situations where some students and faculty have up to date software and machines and other students and faculty do not. This causes problems. If some students have Excel and Mac-Plus and the faculty and other students have Multiplan and the 128 or 512 there is an imbalance. Such differences between students can result in some students receiving higher grades than other students, not because of superior intelligence, study habits, but because of having a better machine and turning out a superior paper. There is also the possibility that faculty will abandon the computer and go back to traditional methods of teaching as they find it financially impossible to keep up with their students technologically. They may also feel computer intensive education is not worth the candle, if they witness new machines and software that is made available to faculty allocated in a way where those receiving the resources are not those who are using them.

There is also the question of whether funds and time spent on computer integration have been beneficial to students and faculty. Much of the research on traditional computer assisted instruction show that benefits are usually of a temporary nature and could have been similarly obtained by having instructors and students spend additional time and effort using the traditional mode.

Some Good Things

According to another survey conducted of Drexel students, the adoption of the personal computer enhances Drexel prestige and helps Drexel attract more students, especially good students. Students are generally excited about the Macintosh and happy with computerized teaching. As time passes, more students value the Macintosh as a learning tool. Faculty members have benefitted from the word processor which enabled them to expedite their writing and the various computing and graphic software which gave them additional research tools.

Rapid Change in Technology

Adoption of the Macintosh, as mentioned above, was not without difficulties. Some have been overcome and some still exist. The rapid change in technology and the intensive competition in the personal computer business result in the constant flow of new models and the quick obsolescence of any computer. Macintosh is no exception. Drexel students paid $1095 for a 128K Macintosh in the fall of 1983; they paid $1020 for a 512K Macintosh in the fall of 1965; and the 1986 freshman are given Macintosh Plus for $1390. Some of those owning old models of the Macintosh are upset by the fact that many new pieces of software cannot be run on old Macintosh. This problem may have been averted had the administration foreseen the problem and negotiated with the manufacture for dealing with this problem.

Conclusion

There are the other problems of change; the technology has had a direct impact on the way we do things and the way we perceive others. The problems and successes discussed in the paper are the more obvious. The subtler consequences of the move to the Mac are many, varying from clerical staff workloads, confilcts between established and emerging centers of authority, and faculty-student relations. Unfortunately many of these are not even perceived, yet alone acted upon. And this is sad, because in the end it may be the way these subtler, very human, consequences are dealt with that will mark the experiment.

On the other hand the employment of the Macintosh has been exciting, seeing students involvement in a new kind of writing system that combines pictures and words convinces us that the computer can be as important in the history of communications as the printing press. The Mac, whose adoption was originally criticized by some of our faculty IBM fans, has been influential in the computer industry with many of the innovations being adopted by other companies. The adoption of the Mac was especially important in a department which was fully tenured, mostly composed of full professors, and old. Problems, not withstanding, the future of economics education at Drexel promises to be more challenging and innovative than in the past simply because a new way of doing things has caused a rethinking of the content we had been offering and the way in which it was being imparted. Although problems occur, mistakes are made, and aggravation develops; this is part of the educational process. After many years of teaching, people often tend to take a more comfortable route in carrying out their teaching responsibilities. The Mac disrupted the comfort and made teaching surprising, exciting and a process of self-discovery. Whether or not this process will continue is open to question. In some ways the constant technological change has had some positive impact on this process, but whether the costs outweigh the benefits is not known.

COMPUTER ASSISTED INSTRUCTION AND EXAMINATION

David M. Sherman, B.A., LL.B., LL.M
Consultant
Computer Education Facility
The Law Society of Upper Canada
Osgoode Hall
Toronto, Ontario, Canada M5H 2N6

Abstract

Students attending the Ontario Bar Admission Course take two courses taught by computer assisted instruction. A course on Canadian income tax law is made available to students during their articling year, and they access this course from their homes, their offices or Law Society terminals throughout the province. During the Bar Admission Course's teaching term, an accounting course is taught. Students take this course on their own time and complete an exam at the computer. The exam is programmed so that every student gets a randomly chosen different exam of equivalent difficulty. Up to 1,000 students a year take the courses, which reside on a central minicomputer running UNIX. Additional courses are under development, both for the Bar Admission Course and for continuing education for practicing lawyers.

The software is easy for people with no computer background to use, yet it allows them control over what they are doing at any time.

CAI has been generally well received by students. In addition, a number of lessons have been learned regarding the design of CAI. The basics of course authoring require only a simple set of tools, yet these must be used wisely in order to make the material challenging, but not frustrating, to the student.

Background

The Law Society of Upper Canada, the governing body of the legal profession for the province of Ontario, runs the Bar Admission Course (BAC). Following law school, students wishing to become lawyers in Ontario must enroll in the BAC, which consists of a one-year *articling term* and a six-month *teaching term*. Articling consists of working for a practicing lawyer anywhere in the province. The teaching term, which runs from September 1 to February 28 following articling, consists of lectures and seminars, one course at a time, with an exam approximately every two weeks. The teaching term runs simultaneously in Toronto (approximately 800 students), Ottawa (200 students) and London, Ontario (100 students). Successful completion of the teaching term exams entitles a student to be called to the Bar of Ontario.

The Law Society's Department of Education, of which the BAC is a part, also runs an extensive series of continuing legal education programs for practicing lawyers. These cover every area of law and are typically one-day lecture programs featuring speaker panels.

The Law Society has sought for some time to turn the BAC teaching term into more of a "skills training" program and less of a course for delivering basic legal knowledge and rules. However, the lack of uniform background in law school graduates has forced the BAC to address itself to the lowest common denominator in each subject. Since students completing the BAC are authorized to practice in any area of law, the Law Society has always felt it necessary to offer basic instruction in all areas of law. Teaching "skills" at the BAC would therefore require reducing the time spent on the "core" material in some subjects. Computer assisted instruction (CAI) was seen as a possible solution. CAI can be used to provide basic instruction for those students who need it, allowing them to reach a minimum level of legal knowledge.

CAI began at the Law Society in the spring of 1983. A one-year experimental project was undertaken to develop a basic course in income tax law, to be taken by students *during their articling year* in preparation for the teaching term. Students with a strong background in tax, either from law school courses or from articling, would not need to take the course. The teaching of income tax at the BAC teaching term would assume a basic understanding of tax law and teach, in a more practical vein, the *application* of tax law when dealing with clients' problems.

The initial project was successful; and in the spring of 1984, the Society approved creation of a permanent Computer Education Facility with the mandate to develop and deliver CAI both to BAC students and to lawyers as continuing education.

Hardware and communications

All courses reside on a Perkin-Elmer 3220 minicomputer running the UNIX (see footnote 1) operating system. During busy periods, the system typically supports 20 to 30 students at the same time. Student terminal rooms have been set up in Toronto (13 terminals), Ottawa (5), and London (3), in addition to single terminals in law libraries in nine other Ontario cities.

The system can be accessed by modem, either by a direct phone call within the Toronto area or via the Datapac X.25 network. Local Datapac nodes are available in 28 Ontario cities. The communications cost of calls from these nodes is paid for by the Law Society. As well, students articling at large law firms in Toronto and Ottawa

---

[1] UNIX is a registered trademark of AT&T

can reach the system through legal research terminals set up in law firms by QL Systems Ltd., using QL's private X.25 network.

No supervision or human assistance is required for students. The system is designed so that students who have never even touched a keyboard can use the terminals on their own.

Courses offered

Income Tax

The first Computer Education Facility course to be developed was Basic Income Tax Law. This course takes 10 to 20 hours to complete and is designed to give the students a basic understanding of the Canadian income tax system, both personal and corporate. It has been made available to BAC articling students each year since 1985. Students are told that while the course is not compulsory, the material covered in it is prerequisite to an understanding of taxation issues taught in the teaching term. As an incentive to complete the course before the teaching term, students are given free access to the course only until August 31, after which they must pay a nominal fee for connect time. Approximately 700 of the 1,100 students take the course each spring and summer.

The Income Tax course is also used by several law schools and is now being made available to lawyers as a continuing legal education program.

Accounting course and exam

Accounting in a Law Office was taught by conventional lecture and seminar methods through 1984-85. In 1985-86, this course was replaced by an equivalent CAI course. The purpose was to free up two weeks of class time for a new course on practice skills. Students now do the Accounting course on their own time, either before or during the teaching term. The course takes about 25 hours, of which 10 to 15 are spent at the computer terminal; the rest are spent doing workbook assignments.

All students must complete an examination taken at the computer. The exam can only be taken from the Law Society's terminals (not from a student's home or office). Every student receives an exam that is different from that of every other student, but of equivalent difficulty. Pseudo-random numbers are used to shuffle question order and question choices and to change the numbers and the facts of each question. The exam has a time limit (75 minutes), and the amount of time left is displayed on the screen with each question. At the beginning of the exam, each student has up to 15 minutes as a "reading period" to review the questions without answering them (this parallels the arrangements for conventional written exams). When answering questions, the student is asked to confirm each answer before it is accepted; the student can also return to any previously answered question to change or confirm the answer.

The exam was taken by 1,100 students in February 1986. While the initial failure rate was relatively high for the Bar Admission Course (19 students), the mean and median marks were also higher than for other courses. All students who failed on their first attempt passed the exam on a second try. (Supplementary exams are permitted in all BAC courses; the overall BAC pass rate usually exceeds 99 percent.)

Evidence

A set of quizzes on the law of evidence is being completed for student use in December 1986. The course materials will be in paper form; the quizzes will be available as a mechanism for students to test their knowledge before writing a conventional exam. Students taking Basic Evidence Law (which can be chosen as one of a number of available courses) will use the CAI quizzes in place of the conventional lectures used in other courses.

Advanced tax courses

Courses are under development and close to completion in a number of advanced tax areas. These are: Tax Aspects of Family Law, Tax Aspects of Real Estate Transactions, and Tax Aspects of Advising the Small Business. These courses will be offered to lawyers as continuing legal education programs, to be taken at their convenience from their homes or offices.

Software structure and interaction environment

The description below is of the Basic Income Tax course. With trivial differences, it applies to the other courses as well; in fact, all of the courses are compiled from the same source files. All of the software is written in the C programming language.

Course structure

Since many law students have never touched a terminal before, the program has been made as easy to use as possible, while still allowing the student to control his or her progress through the course as well as the material to be covered.

Each course is divided into a number of **lessons**, and each lesson into a number of **modules**. At the end of each lesson is a **quiz**, covering all of the material taught in the lesson.

Each module of each lesson exists in two forms: detailed or "primary" instruction; and a "refresher" form. The primary instruction module contains textual information interspersed with questions every few paragraphs. Many of the questions are not designed to be difficult so much as to ensure that the student is absorbing the material presented. It becomes impossible to "skim" the primary material, as one is forced to stop and answer questions on a regular basis.

The student selects a lesson from a menu and then, following a brief description of the lesson, chooses whether to take it in primary mode or refresher mode, or to go directly to the quiz. In primary mode, each detailed module is presented in succession; the assumption is that the student wishes to see the material. In refresher mode, the student chooses the desired modules from a menu.

Each refresher module consists of a single screen (maximum 23 lines) that summarizes the basic points of the module. Students can use this screen to decide whether they need to review the topic in detail by taking the primary instruction. After each refresher screen is shown, students are asked whether or not to go into the detailed instruction for that module.

At the end of each module, whether in refresher or primary mode, the student is offered the option of seeing the statutory references to that module. The material is taught without reference to specific provisions of the Income Tax Act, as constant reference to the statute adds complexity to the explanation of the tax system. Some students, however, wish to read the relevant section of the Act. These students are thus able to refer to the statutory references at the end of each module.

The Calypso authoring language

The primary modules, and some of the lesson quizzes, are written in Calypso (see footnote 2) a simple, interpreted shorthand for specifying CAI questions.

Calypso scripts contain text and question blocks. Each question block contains: a question; one or more "right answer" specifications; zero or more "known incorrect answer" specifications; zero or more general hints (to be offered if the student's answer does not match any known correct or incorrect answer); and an optional failure message (to be printed if the student has not given the correct answer after a number of tries).

Specific messages can be tied to correct and incorrect answers. Thus, where the student enters an answer that is wrong in a way that the course author has predicted, a response can be given that is tailored to that incorrect answer. Rather than a simple "Wrong. Try again," the student will see something like "Wrong. Aren't you forgetting xyz? Try again." In a subject such as tax, where questions tend to have numerical answers, it is possible to design questions such that any of of 20 different incorrect answers reflects a specific error or combination of errors in the student's reasoning. The personalized answers supplied give the feeling of an individual tutor who can focus on exactly what the student is doing wrong.

Calypso is quite simple--it is a notation rather than a programming language. It can be mastered in about half an hour. While its capabilities are limited, Calypso provides the basic features needed for highly effective CAI when used in conjunction with the overall control software described above. A number of our courses have been written directly in Calypso by non-programmers.

Control by the student

As mentioned, student choice of material is handled by menus. We wished, however, to make it possible for students to stop what they were doing at any time and do something else. This is easily implemented in UNIX by trapping an interrupt character.

For the interrupt character, we chose ESC; which is mnemonically easier for students to remember than, say, DEL or ctrl-C; we tell them that they can press ESC at any time to "ESCape the clutches of the program." Pressing ESC produces a menu that allows students to move to the next module, select a new lesson or a different course, leave a comment on the computer, or quit.

The combination of menus and a single, general-purpose interrupt has proven effective in an environment where instructions must be kept simple, yet students must be allowed to choose at any moment precisely what they want to do.

Course refinement through error logging

When a user progressing through the primary instruction enters an answer that is neither correct nor a known incorrect answer (for which a specific response has already been designed), that answer is recorded in a log file. By examining the log file, the course author can quickly discover all of the unexpected answers that students give. With a little creativity, it is usually possible to divine the particular errors in reasoning (or, in some cases, arithmetic) that led to each wrong answer. It is then a simple matter to edit the relevant Calypso script and insert a new "catch" for this answer. Constant refinement in this manner produces, over time, a program that appears to be extremely "intelligent," for it responds sensibly and usefully to virtually every answer a student tries.

Comments and student messages

A student who has a question about the material and wants human help can press ESC and leave a comment at any time. Comments are also solicited at the end of every student's session.

Where a student has asked a question, the course author will leave a reply on the computer, available as electronic mail the next time the student signs on. As well, minor errors or mistakes in the course, and even material that needs rephrasing to clarify its meaning, can be pointed out by students and fixed immediately by the course author. Students derive great satisfaction from knowing that their suggestions are listened to and implemented, and the course author thus has a constant check on the quality of the materials.

Centralized vs. distributed processing

We are often asked whether our software runs on the IBM PC, and why we cannot simply send a disk to those who want to take our course, instead of requiring on-line connection.

The decision to use a centralized system rather than stand-alone microcomputers was a very conscious and deliberate one. Our reasons were several:

1. Canadian income tax law changes regularly. Some aspects of the course needs revision virtually every month. Retaining the course on our central system ensures that all students are working with up-to-date material.

---

[2] Calypso was obtained in its original form from St. Olaf College of Northfield, MN. It was substantially changed and extended at the Law Society.

2. The ability to gather **comments** is valuable, both in order to improve the course and to respond to individual students' concerns and questions.

3. We can use the **log files** to gather information on unexpected answers, and thus modify and improve the course based on previous students' sessions.

4. We can **control** who uses the course, monitor usage, gather statistics on the amount of usage and length of sessions, and charge for actual usage if need be.

5. We are **accessible** to any brand of microcomputer terminal, provided the user has a modem. In addition, the cost per workstation of terminals is lower than that of microcomputers.

6. The UNIX operating system gives us flexibility unavailable in microcomputers: to run multiple processes per user, to permit communication between them via pipes, and to gracefully handle various kinds of signals (user-generated interrupts, hangups, timeouts and interprocess signals).

Some of the features above could be handled with a properly set up network of microcomputers. At the present time, however, a central system is the only way to satisfy all of the above needs.

Evaluation

Student reaction

Reaction from students taking the income tax course, as expressed through their on-line comments, has been overwhelmingly positive. The ability of the computer to deal with each student's answers on an apparently individualized basis is particularly appreciated by many students. In addition, the fact that students are forced to answer questions is welcomed as a means of encouraging them to focus on the material.

A number of students specifically commented that they were afraid of the computer before they used it, but that they had enjoyed their CAI sessions and consequently felt more positive toward using computers in general. CAI thus has a partial side effect of increasing computer literacy among law students and reducing what might be called "technophobia."

At the same time, there is a group of students who do not find CAI to be effective for them. Some of these students complain that a book would be just as effective; others do not leave any comments on line, but indicate on course evaluation forms that they did not find the computer-taught courses particularly enjoyable. We have come to recognize that CAI, while valuable for many, is not the right tool for some. Accordingly, we make a written transcript of the course available for those who prefer conventional study methods.

The hard-copy issue

Our original feeling was that to make a printed course transcript available would defeat the purpose of offering CAI. Students would simply read the transcript and deprive themselves of the opportunity of interacting with the computer, which could guide them as they made mistakes. This is true to some extent: Some students will not use the terminal unless they feel they are forced to, and yet once they do, they will discover how much more valuable interaction can be than simply reading.

On the other hand, we have concluded that we should not be overly paternalistic toward our students. Bar Admission students have all completed at least five years of university-level education (most, in fact, have completed six or seven). They are mature enough to make their own decisions as to the best form of instruction for them.

In addition, if we refused to make a printed transcript of the full course available, students with their own microcomputers and printers would simply run one off and sell it to other students.

Furthermore, many students who have taken the course on-line have commented, quite validly, that because it covers a substantial amount of material, they should be able to keep a hard copy for reference purposes, rather than having to take extensive notes from the screen.

Accordingly, we have opted for a route that encourages students to at least try the computer, without forcing them to do so. We produce a printed transcript and sell it to the students, but do not make it available until September of the teaching term each year. (The Business Law course, the principle course that requires basic tax knowledge as prerequisite, is taught in October.)

We also make the transcript available to pregnant students who have fears about using video display screens. While we reassure these students that all available medical evidence indicates that there are no problems associated with pregnant women using terminals for a few hours, we do not force them to use the terminal if they do not wish to. For the Accounting exam (the only part of the CAI system that requires student use of terminals), we permit such students to do the exam on paper if they wish (their answers are then typed into the computer by BAC staff).

Comments on CAI authoring

As was noted above, the Calypso authoring language, which we have used for the primary modules, is quite simple. In fact, only a very basic set of tools is needed to produce highly effective CAI. Several points can be noted.

First, the computer must not be allowed to become an "electronic page turner." That is not CAI. It is crucial to pepper textual information with lots of questions to keep the student active rather than passive. Even simple questions, which do little more than confirm that the student has read the previous paragraph, have a tremendous reinforcing value.

Second, questions should be designed so that guessing is discouraged. Multiple-choice and yes/no questions invite the student to say, "Oh, let's try this answer and see what happens," particularly when there is no penalty associated with wrong answers. While the occasional multiple-choice question is acceptable, the bulk of the questions should be "open-ended"--that is, they should ask for a number, a keyword answer, or even a phrase. The student is thus forced to think about the problem and come up with a possibly correct answer, rather than blindly guessing out of mental laziness.

Third, where open-ended questions are used, it is vital to provide useful responses when students' answers are incorrect. Ideally, all possible wrong answers should be predicted and dealt with individually, focusing on the exact mistake each student has made. Where students' answers were unexpected, general hints should be provided that may assist them in reaching the correct answer. Nothing is more frustrating to students than the simple words "Wrong. Try again."

Fourth, students should be allowed a reasonable number of attempts at each question. The number of retries allowed will depend on the question. Obviously, only one chance should be allowed for a yes/no question. For open-ended questions, there is a fine line between too few retries (not giving students the chance to work out the answer on their own) and too many hints and retries (causing frustration).

Future work

CAI is here to stay at the Law Society. The courses on income tax, accounting, and evidence will be maintained; and new courses will be developed over the next few years. As the development of continuing legal education courses progresses, we expect to see substantial use by lawyers. The computer may come to play a major role in delivering continuing education to the Ontario legal profession.

We continue to grapple with the choice of subject matter for CAI courses. It is apparent that the subjects best suited to CAI are those in which the law to be taught is complex yet clear. Tax is particularly appropriate because it lends itself to questions with numerical answers, which require the student to figure out, rather than guess at, the correct answer. It appears that many other subjects can be usefully taught by CAI as well, however.

We are also exploring the development of simulations. These would be much more complex programs, modeling the development of a legal case or file. The student, playing the part of a practicing lawyer, would be required to make decisions as to the steps to take, and the effects of those decisions would be felt throughout the rest of the simulation. Subject areas we are considering in this regard include corporate/commercial law, family law, and real estate transactions.

TEACHERS, COMPUTERS AND CURRICULUM GOALS: AN INSERVICE VIDEOTAPE FROM DOWN UNDER

Betz Frederick
Harold Frederick
University of Illinois
University High School
1212 W. Springfield Ave.
Urbana, IL 61801

Abstract

Use of computers in schools can be a powerful tool but integration into the school curriculum doesn't just happen. It must be a planned event. Unless teachers are given professional development, the introduction of computers into schools will not have the effect on education that is possible. Teachers want and need to be shown examples of how other teachers are using computers in their teaching.

A videotape was developed to give viewers an impression of what happens in a teaching situation where computers are being used. It shows a variety of different teaching situations. Teachers in the videotape demonstrate how computers make their teaching more effective, easier, fun and interesting. They give personal reasons as to why they continue to use computers with their classes. In this way, teachers can draw upon the experience of other teachers.

The video leads teachers to focus their attention on areas of their curriculum that can benefit from the use of computers and to devise ways of using computers effectively to satisfy their own curriculum goals.

Introduction

The call to introduce computers into schools has been coming from society at large and broadcast through the media over the past couple of years. A survey conducted by Schuttenberg, McArdle, and Kaczala (1985) indicated that parents, educators and business leaders felt that computers would be very important in the future and it was essential that students learn to use them. They can also be very effective tools in the teaching/learning process. Evidence reported by Hall and Jamison (Hathaway, 1984) indicated that computer-based education produced as much, and often more, learning than traditional methods of instruction. Lipson and Fisher (1983) wrote "We believe that schools can effectively incorporate computers into the existing system. We suspect that the computer and the videodisc technologies will prove to be the most valuable teachers' aides ever invented."

According to Moskowitz and Birman (1985), an essential ingredient to the successful use of computers in schools is trained teachers and there is no shortage of teachers interested in receiving training. Van Lengen (1985) argues that, "No matter how good the technology is, it still takes trained people to conceive ways to use them. The people must not only want to use the new technology, but also they must be properly trained in its use."

Hugh Burkhardt, (1986) from the Shell Centre for Mathematical Research, Nottingham University, U.K., states that for any innovation to succeed on a large scale it must:

1. make the teacher's job easier
2. be fun/interesting
3. help with a known problem
4. have a measure of public support

It is already known that computers have a measure of public support. It is the responsibility of teachers to find ways that they can use computers to make their teaching more effective.

The authors' experiences have shown that if teachers identify ways in which the computer could be used 1) to make the teacher's job easier, 2) to help with a known problem and 3) to make lessons fun or interesting, they would be better able to devise ways that the computer could be integrated into their own curriculum. The use of these three focus questions proved to be effective during a computer inservice led by one of the authors (Frederick, 1984). The teachers identified 14 subject areas and 78 topics where the computer could fit into the curriculum. Every department in the school identified some ways in which they could use computers.

Visits or Videos?

The authors' personal experiences with teachers in inservice courses indicate that the participants need and want to be shown examples of how other teachers are using computers in their teaching One way to satisfy this need is to arrange visits to schools to see computers being used by students. The advantage of such visits is that participants are able to see first hand, in unrehearsed situations, how computers are being used in a few curriculum areas. It would be impossible for all teachers to visit schools to gain this knowledge for themselves because of the burden it would put on the small number of schools that have successfully integrated the computer into the various subject areas. In addition, the amount of travel time, and timetable coordination necessary would be prohibitive. Teachers would probably only be able to see computers in one or two classrooms.

Videotape is another method of giving the teachers experience with how computers are being used across the curriculum. Moss (1983) reports that videotape materials are a "peculiarly appropriate method of solving the problem of how to cope with the unprecedented need to retrain teachers". Moss goes on to state that videotape materials can be designed to include examples of applications relevant to a specific part of the curriculum.

An Inservice Videotape

The authors felt that a videotape which shows how computers can be used across the curriculum would be a valuable resource for professional development in computer education and have widespread use. Therefore the videotape was produced. It can be used by tertiary level institutions for preservice education, by teacher consultants for inservice courses, by resource teachers for staff development and by any individual interested in using computers across the curriculum. Teachers can view it during their own time, without leaving their school and see computers being used in many different ways. Consultants can forward it to a school before an inservice so that the teachers would have a better understanding of how computers could be integrated into the curriculum. It can be used to stimulate discussion during an inservice, and viewed as often as necessary after the inservice.

After watching the videotape, teachers should:

1. have increased confidence and think "I can do that",
2. be able to identify ways they can use computers as a tool in teaching,
3. and have learned various reasons why other teachers use computers with their classes.

The videotape presents general ideas that help teachers to identify ways that they can use computers to achieve their own curriculum goals. The videotape presents information and provokes thought about general issues of educational computing. The videotape shows a variety of different teaching situations to help broaden the viewer's concept of instructional computing. Teachers in the videotape demonstrate how computers have solved a known problem for them, made their teaching more effective, easier, fun and interesting. They give personal reasons why they continue to use computers with their classes.

The integration of computers across the curriculum is a difficult task, but hopefully with the use of the videotape, teachers can draw upon the experience of other teachers, and devise ways of using computers effectively to satisfy their curriculum goals.

This videotape can be purchased for $50.00 from the authors. To order or get more information write to: Betz Frederick,
1504 Gates Place, Champaign, IL 61820.

References

Burkhardt, H. "Debate: The Mocrocomputers: Miracle or Menage in Mathematics Education". In M. Carss (Ed.), Proceedings of the Fifth International Congress on Mathematical Education, 359-372, Boston, Birkhauser, 1986.

Frederick, B. R. A Case Study Evaluation and Report of a Computer Inservice Program at Catherine McAuley Girls High School. An unpublished report written in 1984.

Hathaway, M. "Variable of computer screen display and how they affect learning", Educational Technology, 24, 1, January 1984, 7-9.

Lipson, J. T. and Fisher, K. M.; "Technology and the Classroom: Promise or Threat?" Theory Into Practice, 22, 4, 1983, 253-258.

Moss, R., Video the Educational Challenge. Great Britain, Biddles Ltd., 1983.

Moskowitz, J.H., and Birman, B.F.; Computers in the schools: Implications of Change", Educational Technology, 25, 1, January 1985, 7-14.

Schuttenberg, E. M., McArdle, R. J., and Kaczala, C. M.; "Computer uses in schools: research on what is and what should be", Educational Technology, 25, 4, April 1985, 19-22.

Van Lengen, C. "Will we effectively use the computer as a tool in education?" Educational Technology, 25, 1 January 1985, 47-48.

Retraining Teachers in Low Demand Disciplines as Computer Learning Teachers*

Gary G. Bitter, Ph.D.
Richard Sheets

Arizona State University
Phoenix Union High School District

Abstract

The Retraining Teachers in Low Demand Disciplines as Computer Learning Teachers was designed with the following objectives:

1.0 To increase the retention of experienced teachers in the teaching profession by retraining them into a high demand discipline, computer learning, from low demand disciplines.
2.0 To increase the available pool of teachers of computer learning.
3.0 To attract qualified teachers and give them an incentive to retrain rather than leave the profession by paying for all course tuition and a stipend.
4.0 To develop a model program that can be used to retrain teachers in other disciplines into high demand disciplines like mathematics, science, and computer learning in cooperation with local school districts.

The project was funded by a grant from The Education for Economic Security Act. Each participant received tuition waivers for all courses and $600 upon completion of all project requirements. In addition Apple Computer loaned each participant an Apple IIc computer, color monitor, and printer for the duration of the project. The project retrained nineteen teachers who had previously been involved in local school districts reduction in force criteria. The teachers completed fifteen semester hours of computer based education classes during the Academic Year 1985-1986 and summer 1986. In addition each participant completed fifty hours of internship in computer related experiences. The Fall of 1986 found all teachers employed full time with two of the teachers being RIF'd and finding employment as computer learning teachers in other school districts. The project indicates that teachers in low demand areas can be retrained into a new academic discipline through an intensive program with financial incentives.

Introduction

Teachers of computing skills are in high demand as computer technology continues to infiltrate modern society and increasingly becomes an important part of our service-oriented economy. Conversely, many highly skilled and experienced teachers across the country in low demand disciplines such as Physical Education, Home Economics and Fine Arts are being laid off (referred to as Reduction In Force, RIF, in this paper) due to declining enrollments. Many of these teachers may be unable to find work in other school districts in their present disciplines due to a glut of teachers available in their fields with less experience and therefore lower salary expectations. This represents a substantial loss to our society and educational system of years of hard-earned pedagogical experience and skills.

In recognition of the critical shortage of qualified computer learning teachers, The Arizona State University College of Education, in conjunction with the Phoenix Union High School District, and funding from the Education For Economic Security Act developed and field-tested a model for retraining classroom teachers of other disciplines to become teachers of computer learning.

Objectives of Project

1.0 To increase the retention of experienced teachers in the teaching profession by retraining them into a high demand discipline, computer

*Funded by a grant from the Education for Economic Security Act

learning, from low demand disciplines.

2.0 To increase the available pool of teachers of computer learning.

3.0 To attract qualified teachers and give them an incentive to retrain rather than leave the profession by paying for all course tuition and a stipend upon completion of all courses and project requirements.

4.0 To develop a model program that can be used to retrain teachers in other disciplines into high demand disciplines like mathematics, science, and computer learning in cooperation with local school districts.

Selection of Participants

The retraining project selected thirty-two participants to enter the training. The selection process was based on the teacher having previously been involved and the likelihood of the teacher to continue to be involved in a Reduction in Force (RIF). Twenty-five of the participants were funded through the grant while eight paid their own expenses with the agreement that they would have the option to replace any of the original twenty-five who withdrew from the program. The eight were rank ordered by qualifications for replacement. The project retrained twenty teachers from the Phoenix Union High School District, a declining enrollment district that has experienced several Reduction in Forces of experienced personnel in low demand disciplines, and five teachers "at large" from other local public and private schools, to teach computer learning.

The participants were elementary and secondary teachers. These included five special education teachers, two band teachers, two physical education teachers, six English teachers, three home economic teachers, five business teachers, three math teachers, one science teacher, two industrial arts teachers, two foreign language teachers, and one social studies teacher.

Academic Retraining

The students enrolled at Arizona State University in a three semester hour Computer Literacy and Applications course and a three semester hour Computer Programming course during the Fall of 1985. In the Spring semester the participants enrolled in a three semester hour Introduction to Data Processing course and a three semester hour Introduction to Pascal class. The final course offered for the participants was held in the summer of 1986 and was a three semester hour Intermediate Pascal course. In addition all students were required to participate in fifty hours of internship. This internship included attending conferences, participating in training sessions, assisting in teaching computer classes or individually helping teachers in using computers.

Computer Equipment

Each participant was provided an Apple IIc with a color monitor and printer by the Apple Computer Corporation. This convenience allowed each participant to do assignments at home as well as try out ideas in their classes. In addition each student was provided Appleworks, Pascal, Apple Logo and all necessary DOS and BASIC software.

Final Results

The project was completed by nineteen teachers who each received a $600 stipend. Several of these teachers are now teaching computer classes but most have expanded their role as a classroom teacher to include computer usage. Three of the teachers were RIF'd but found computer related positions in other school districts. All teachers were rehired or found computer education positions outside of their school districts. An extensive follow-up of all participants will be done in June, 1987.

Conclusions

The participants were intensely prepared to utilize computers in education. The project was highly successful in maintaining jobs for all participants who completed the project and several are now employed as full time computer education instructors while most of the others are utilizing computers extensively in their regular instruction. The project has shown that experienced teachers in low demand disciplines can be retrained into a new academic discipline through an intensive academic program with financial incentives and continue to be valuable members of the educational community.

PENNSYLVANIA'S PROGRAM FOR INSERVICE TRAINING FOR TEACHER'S IN COMPUTER-ASSISTED INSTRUCTION

Anne Weiner
Associate Dean of Academic Affairs

Matthew Bodek
Assistant Director, Regional Computer Resource Center

Philadelphia College of Textiles and Science
Schoolhouse Lane and Henry Avenue
Philadelphia, Pennsylvania 19144
(215) 951-2835

Abstract

This paper discusses the Information Technology in Education for the Commonwealth (ITEC) program. To meet the challenge of increased microcomputer use in education, the State of Pennsylvania responded with a program to provide inservice training for teachers in microcomputers and in computer-assisted instruction.

A major feature of the program is a network of 14 Regional Computer Resource Centers (RCRC) located at universities, colleges, community colleges, and state intermediate units across Pennsylvania. Each facility is comprised of a microcomputer classroom and a software library. The microcomputer classroom allows hands-on training while the software library is available for interested teachers to evaluate the use of microcomputer courseware in their classroom.

Each RCRC offers tuition-free graduate courses in the educational applications of microcomputers at the elementary and secondary school level. After completing the course, teachers will be able to evaluate, select, and recommend for purchase microcomputer hardware and software to meet classroom needs; use utility courseware; demonstrate a familiarity with LOGO or BASIC languages; and recognize the implications of microcomputer use in schools.

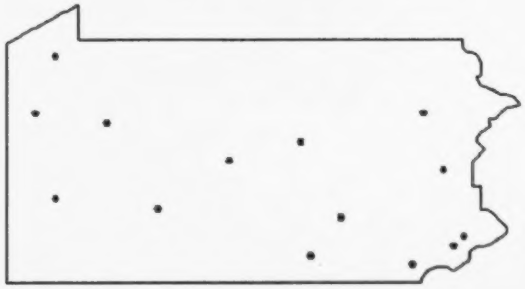
The statewide coordinator for the program is the Teacher Education Center (TEC) located at Clarion University. Primary fiscal, administrative, and planning responsibility is assumed by the Pennsylvania Higher Education Assistance Agency (PHEAA). The program's future will continue to emphasize teacher training. The program will eventually raise microcomputer literacy in Pennsylvania's schools.

The past several years have seen a rise in the use of computers in elementary and secondary classrooms. Computers in the schools have moved from the computer labs designed to facilitate instruction in computer science to the use of computers in every classroom to aid instruction in subject areas. This increased use of computers in the classroom has presented several problems for school administrators and other education leaders. Decision regarding choice of hardware and software have long-range budget and usability consequences. Adequate facilities to house and secure equipment must be identified. Teachers must be encouraged to use the equipment and use it effectively in their individual classroom and programs.

Many teachers are unfamiliar with computer operation, unsure of the advantages of computer use, and unable to develop realistic and effective strategies for computer-assisted instruction. Educators in the state of Pennsylvania realized this situation and worked with state legislators to develop and fund a program to provide inservice training to teachers in microcomputers and computer-assisted instruction. This paper describes the program with the hope that successful elements can be used by others interested in teacher inservice programs for computer education.

The Information Technology in Education for the Commonwealth (ITEC) program is comprised of 14 Regional Computer Resource Centers (RCRC) located in key areas across Pennsylvania.

Location of Pennsylvania's
Regional Computer Resource Centers



The goals of the (ITEC) programs include: encouraging orderly planning for the use of microcomputers and the application of microcomputers to instructional programs of elementary and secondary schools; improving teacher training in computer education; encouraging schools and school districts to acquire computer hardware; and assisting them in the acquisition of appropriate computer software.

The RCRC network is a linkage of seven universities, two colleges, one community college, and four state intermediate units. Center sites were selected geographically so that most teachers would not have to drive more than 50 miles to a center. While each of the 14 centers is unique in location and clientele, the activities of the center reflects the overall program goals. These activities focus on presentation of courses and workshops, review of hardware/software grant proposals, development of software preview libraries, and a provision for educational advisory services. RCRC sites are connected by a telecommunications center and this communications network provides immediate access to communiques and announcements, exchange of electronic mail, and innovative ideas and creative programs.

The facility of each center includes two primary areas: a software library and preview room; and a microcomputer laboratory/classroom. The library provides access to educational software for teachers who are interested in evaluating, selecting, and incorporating microcomputer software in their curriculum. Additional resources, such as the TESS and EPIE indices, are available to assist teachers in evaluating software. Several of the centers have also developed a collection of public domain software, and many have examples of advanced educational technology such as interactive video.

In addition to microcomputer software, the library contains information that may assist interested teachers in establishing microcomputer facilities in their schools. Grant information and technical assistance is also available. Literature on security systems for both software and hardware is also available at each center.

The microcomputer laboratory/classroom complements the software library as it facilitates hands-on training in word processing, data-base management, spread sheets, and programming. The microcomputer labs in the centers are equipped with machines from the Apple II family. Other types of microcomputers are also available in the center.

The ITEC courses are a major feature of the program. Tuition-free graduate courses in the educational application of computers are offered for elementary and secondary teachers. The syllabi for the ITEC elementary and secondary courses include activities which help reach the following objectives: evaluate, select, and recommend for purchase software and hardware to meet classroom needs; use popular utility programs involving word processing, data base management, spread sheet, electronic gradebook and print utilities; demonstrate entry level competency in LOGO (elementary school teachers) or BASIC (secondary level teachers); and recognize philosophical, ethical, and legal implications for microcomputer use in an educational environment.

A regular program of workshops addresses identified areas of interest such as the use of computer graphics and specialized software evaluation. These activities are supported by the software library housed in each center.

An advisory board composed of five local educators has been appointed for each center. These boards provide input and evaluation for center activities and review grant applications for the hardware/courseware purchase support component of the ITEC program. During the first year of the program many board members were actively involved in center activities and their continued support and evaluation is sought by center personnel.

Each center provides scholarship funds for teachers to defray the tuition costs of the three credit graduate course described earlier. In most regions waiting lists are maintained as the number of qualified applicants greatly exceeds the number of openings in the courses as determined by tuition funds. Course enrollment selection procedures favor teachers with the least knowledge of microcomputers but the greatest desire to include them in their curriculum and classroom. These courses are recognized by the Pennsylvania Department of Education and may be used by teachers for certification requirements, the master's equivalency, as part of a master's degree program, or as professional enrichment.

The ITEC Teacher Education Center (TEC) at Clarion University is the statewide coordinator for the teacher inservice component of ITEC. TEC personnel facilitate program and staff development, oversee program-wide policies and guidelines, coordinate statewide publicity, collect and analyze ITEC course evaluations, and provide educational assistance and feedback to all RCRC directors and instructors. The Pennsylvania Higher Education Assistance Agency (PHEAA) assumes primary fiscal, administrative, and planning responsibility for the ITEC programs.

Future program activities for ITEC and the centers will continue to emphasize teacher training, software evaluation, and hardware acquisition. Further expansion of the center's activities and services include placing increased emphasis on disseminating information to school district personnel on the effective use of computers in schools and in the classroom and on delineating strategies for the effective implementation of the new educational technologies. Ultimately, ITEC will raise microcomputer literacy in Pennsylvania's schools and make the Commonwealth's teachers self-sufficient in microcomputer education.

CONTINUING EDUCATION ACTIVITIES OF THE ACM

Robert M. Aiken, Chair
Temple University
Philadelphia, PA

ABSTRACT

The Association for Computing Machinery has taken an active role in providing continuing education opportunities for its members and for the general public. In addition to its well-publicized curriculum activities, the ACM Education Board provides other important continuing education services to individuals interested in computer science: it sponsors week-long tutorial programs once or twice each year in major cities; it sponsors professional development seminars for local chapters; it cooperates with the Computing Sciences Accreditation Board in reviewing and accrediting undergraduate programs in computer science; and it publishes question-and-answer articles that allow individuals to appraise and develop their knowledge about different aspects of computer science.

This panel will focus on ACM's Tutorial Week, its professional development seminars, its accreditation activities, and its self-assessment procedures.

PANELISTS

Neal S. Coulter
Florida Atlantic University
Boca Raton, FL

Julia E. Hodges
Mississippi State University
Starkville, MS

Helen C. Takacs
Mississippi State University
Starkville, MS

A. Joe Turner
Clemson University
Clemson, SC

THE PLACE OF COMPUTER SCIENCE IN THE LIBERAL ARTS/SMALL COLLEGE PROGRAM

R. Waldo Roth, *Chair*
Taylor University
Upland, IN

ABSTRACT

Panel members will discuss the institutional position of their colleges in deciding what—if any—type of computer science major to offer. The positions of the panel will range from those who have selected for a variety of reasons not to have a computer science major to one with a full blown program.

Audience participation will be solicited to get a variety of views on this issue, including accreditation, cost, philosophy, curriculum, and implications.

PANELISTS

Thomas Moberg
Kenyon College
Gambier, OH

Alice Simpkins
Paine College
Augusta, GA

Peter Smith
St. Mary's College
Notre Dame, IN

SPONSOR: ASCUE

EXPERT-SYSTEM BASED INSTRUCTIONAL SYSTEMS

M. C. Lee
Department of Computer Science
The Chinese University of Hong Kong
Shatin, N. T. Hong Kong

Abstract

The expert-system based instructional systems described in this paper contain a general purpose expert-system shell. The shell is a suite of logic program modules written in PROLOG, which can be used to construct domain-dependent instructional systems.

An expert-system approach to instruction allows a computer to center on the learner instead of the teacher. It helps students learn concepts, think analytically, and test ideas. The shell creates a versatile way for students to explore knowledge about a particular subject. In addition, it encourages students to learn actively and to think logically.

Introduction

Computer based instruction (CBI) conventionally centers mainly on the teacher. It uses a fixed sequence to drill students about course materials. Often the CBI exercises fail to consider the different abilities of students. The programs predetermine how to handle both questions and answers. Effective teaching requires far more intelligence.

An expert-system based instructional system (ESBIS) uses both an interactive expert-system shell and a knowledge base, which depends on the topic. The shell and the knowledge base are written in PROLOG, a language based upon logic. Specifically, PROLOG implements a subset of first-order logic. Since first-order logic is a relatively simple formalism and has been well developed for a long time, knowledge representation in PROLOG is easy to grasp. In general, PROLOG expresses course materials in clauses of the form:

C if P1 & P2 & . . . & Pn

A clause is actually an implication in which the conjunction of zero or more conditions Pi implies at most one conclusion C. In syntax, the conclusion appears first; and the conditions appear last, to the right of "if," separated by the "&" or "and" connectives. Both the conditions and the conclusion are the atomic formulae of predicate logic. A clause without the condition part is known as an "assertion" or a "fact," whereas a clause having both the conclusion and the conditions is known as a "rule."

There are now different implementations of PROLOG for microcomputers; the dialogues presented in this paper are based upon micro-PROLOG (Clark, 1984). Its SIMPLE front end allows a readable sentence format.

An Interactive Expert-System Shell

The ESBIS shell is composed of a number of logic program modules such as "verify," "find," "how," "whynot," etc. It allows students to pose different types of queries. The "verify" program allows the student to request verification of ideas, expressed in the form of a conjunction of one or more sentences that follow the SIMPLE syntax. The "find" program enables a question to be posed to the PROLOG inference engine to find all possible solutions. The "how" program allows the student to request information about the way the program deduced its response to a "find" query. The "whynot" program permits students to ask why a specified goal failed when the inference engine evaluated an earlier query.

Users may extend PROLOG and readily upgrade the shell by micro-PROLOG primitives used together with any existing commands. The modules included in the shell can adapt to any knowledge base. The content of the questions does not restrict them. As general purpose logic programs, they work on logic forms rather than on specific content.

Learning Through Interaction Examples

In this section, we elucidate by means of simplified examples, based upon the domains of elementary number theory and set theory in mathematics, how ESBIS can help students learn.

The elementary number theory knowledge base contains many logic programs for defining simple mathematical concepts. Students can experiment and check the validity of what they understand about primitive concepts such as factor, common factor, highest common factor, etc.

In order to test a concept on factor, a student may first add his/her own definition of factor to the knowledge base with the "add" command provided by the SIMPLE front end.

```
student   :   add(x factor-of y if x integer-in-range (1 y)
                    and
                 x product-of (z y)
```

The definition means that x is a factor of y if x is any integer within the range (1 y) such that y is the product of x and a certain number z.

The student can now test his/her definition:

```
student  :  find(x : x factor-of 4)
computer:   answer is 1
            answer is 2
            answer is 3
            answer is 4
            No (more) answers
```

The program may mystify the student when it shows 3 as one of the factors of 4. This unexpected answer may mean that the student's definition is not correct. To trace what has gone wrong, the student can ask:

```
student  :  how(3 factor-of 4)
computer:   To show 3 factor-of 4, I could use the rule

            x factor-of y if
               x integer-in-range (1 y) and
               y product-of (x z)

            I can show:

            3 integer-in-range (1 4) and
            4 product-of (3 1.33334).

            So 3 factor-of 4 confirmed.
```

With the explanation, the student can readily identify how he/she had misunderstood and can modify the definition accordingly:

"x is a factor of y if x is any integer in the range (1 y) such that y is the product of x and a certain integer z."

The previous "factor-of" logic program should also be updated to reflect the modification of the definition as indicated below:

```
x factor-of y if
   x integer-in-range (1 y) and
   y product-of (x z) and
   z an-integer
```

The user can confirm the updated definition by posing a number of queries similar to the one given below.

```
student  :  find(x : x factor-of 4)
computer:   answer is 1
            answer is 2
            answer is 4
            No (more) answers
```

Having grasped the concept of factor, a student can readily define the concept of common factor:

```
student  :  add(x common-factor-of (y z) if
               x factor-of y and
               x factor-of z )
```

It means that x is a common factor of y and z if x is a factor of y and x is a factor of z.

The user can verify the definition by querying the PROLOG inference engine, and comparing PROLOG's answers with textbook answers. For example,

```
student  :  find(x : x common-factor-of 4 14))
computer:   answer is 1
            answer is 2
            No (more) answers
```

The textbook answers confirm the definition.

Similarly, the student can use simple concepts to construct a conceptual definition for the highest common factor (HCF):

```
student  :  add(x HCF-of (Y Z) if
               y isall (z : z common-factor-of (Y Z)) and
               x maximum-of y )
```

The SIMPLE front end provides the "isall" predicate; the knowledge base provides the "maximum-of" predicate. The conceptual definition means that x is the HCF of Y and Z if x is the maximum among the list y of all possible common factors of Y and Z.

Again, the student can experiment with the validity of the definition by posing a series of questions such as:

```
student  :  find(x : x HCF-of (24 36))
computer:   answer is 12
            No (more) answers
```

The student can further experiment with many of his/her ideas about common factors. For instance, the query below asks whether it is true that, for all integers y and z in the range (1 10), if x is a common factor of y and z, then x is a factor of y+z.

```
student  :  verify( (forall y integer-in range (1 10) and
                            z integer-in range (1 10) and
                            x common-factor-of (y z) and
                            SUM(y z y1)
                     then x factor-of y1) )
computer:   answer is yes
```

The SIMPLE front end provides the "forall . . then" construct. The "yes" response from the inference engine idicates that if x is a common factor of y and z, then x is a factor of y+z.

Consider another knowledge base that embodies several conceptual definitions in set theory, including subset, union, intersection, relative complement, absolute complement, etc. With these definitions, students can check their understanding or ideas about concepts relating to the set theory domain.

For instance, the dialogue below attempts to confirm the conjecture that the relative complements of two sets follow the commutative law; i.e., for any two sets A and B, the relative complement of B with respect to A is the same as the relative complement of A with respect to B.

```
student:  add(commutativity-of-relative-complement if
             X universal-set and
             (forall x subset-of X & y subset-of X and
                  Y relative-complement-of (x y) and
                  Z relative-complement-of (y x)
             then    Y EQUAL Z) )
```

The definition means that for all subsets of x and y of X, a universal set, if Y is the relative complement of y with respect to x and Z the relative complement of x with respect to y, then x and y are equal.

To confirm the commutative property of relative complement, the student can pose the query below.

```
student  :  verify(commutativity-of-relative-complement)
computer:   answer is no
```

In case the student fails to understand the negative response, he/she can seek an explanation as follows:

```
student  :  whynot(commutativity-of-relative-
                complement)
computer:   To show commutativity-of-relative-
                complement,
                I could use the rule

            commutativity-of-relative-complement if
                X universal-set and
                (forall x subset-of X and
                        y subset-of X and
                        Y relative-complement-of (x y) and
                        Z relative-complement-of (y x) and
                 then   Y EQUAL Z)

            It can be shown:

            (a b c) universal-set and
            () subset-of (a b c) and
            (a) subset-of (a b c) and
            (a) relative-complement-of (()(a)) and
            () relative-complement-of ((a)()).

            But (a) EQUAL () fails.
```

These explanations should help the student realize that the previous conjecture was wrong. He/she can now conclude that the relative complement does not have the commutativity property.

A set theory lecture might teach a student the transitivity property of set inclusion, i.e., if X is a subset of Y and Y a subset of Z, then X is a subset of Z. If the student distrusts the information or questions whether he/she understood the lecture, he/she can ask:

```
student  :  verify(X universal-set &
                    (forall x subset- of X and
                            y subset-of X and
                            z subset-of X and
                            x subset-of y and
                            y subset-of z
                     then   x subset-of z) )
computer:   answer is yes
```

The query means that for any arbitrary universal set X, if x, y, and z are subsets of X, and x is a subset of y, and y is a subset of z, then x is a subset of z. The affirmative response confirms that set inclusion is transitive, and thus clears any doubt about the lecture.

Evidently, students can use the basic concepts already encapsulated into the knowledge base to experiment with many of their concepts and ideas concerned with the set theory domain. On many occasions, students can readily pinpoint why their conceptual ideas err by making use of the "how" and "why not" facilities provided by the system.

Note that the present system also solves problems. For example, suppose a student has to solve the problem as specified below:

Given a universal set u = (1 2 3 4 5 6 7), a set A = (1 2 3 4 5), and another set B = (2 4 6), find (A/B) ' U B; where A\B is the relative complement of B with respect to A, (A\B)' is the absolute complement of (A\B), and U is the set union operator. In order to find an answer to the problem, the student can interact with ESBIS as follows:

```
student  :  add(x goal if
                X relative-complement-of ((2 4 6)
                    (1 2 3 4 5)) and
                Y absolute-complement-of X and
                x union-of (Y (2 4 6)) )

student  :  update universal-set
computer:   enter new definition :
            (1 2 3 4 5 6 7) universal-set

student  :  find(x : x goal)
computer:   answer is (2 4 6 7)
```

In case the given answer does not match the one found manually, the user can ask ESBIS to explain how it deduced the answer.

```
student  :  how((2 4 6 7) goal)
computer:   To show (2 4 6 7) goal, I could use the rule

            x goal if
                X relative-complement-of ((2 4 6)
                    (1 2 3 4 5)) and
                y absolute-complement-of X and
                x union-of (y (2 4 6))

            It can be shown:

            (1 3 5) relative-complement-of ((2 4 6)
                (1 2 3 4 5)) and
            (2 4 6 7) absolute-complement-of (1 3 5) and
            (2 4 6 7) union-of ((2 4 6 7) (2 4 6)).

            So (2 4 6 7) goal confirmed.
```

A student who does not understand how the program derived the result for the absolute complement, can pose another "how" query as follows:

```
student  :  how((2 4 6 7) absolute-complement-of
                (1 3 5))
computer:   To show (2 4 6 7) absolute-complement-of
                (1 3 5), I could use the rule

            x absolute-complement-of X if
                y universal-set and
                x remove-from (X y)

            It can be shown:

            (1 2 3 4 5 6 7) universal-set and
            (2 4 6 7) remove-from ((1 3 5)
                (1 2 3 4 5 6 7)).

            So (2 4 6 7) absolute-complement-of
                (1 3 5) confirmed.
```

Where "x remove-from (X y)," a predefined predicate, means that x is the result of removing set X from set y. Clearly, a student can request ESBIS, step by step, for the "how" or "whynot" explanations to any level of possible nesting.

Teachers' Roles

Conventional CBI systems necessitate teacher support; so does the present interactive expert-system shell.

A teacher can play a number of roles. For instance, the teacher can prepare the topic-dependent knowledge bases. These should contain all the basic definitions. And for each knowledge base, the teacher should prepare an on-line manual to introduce the predicate names. Furthermore, the manual should also introduce the basic concepts and provide hints as to the range of topics with which a user may experiment.

Concluding Remarks

The instructional systems mentioned in this paper are based upon a general purpose expert-system shell, which provides valuable facilities for various queries such as "verify," "find," "how," "whynot," etc. The shell together with a domain-dependent knowledge base can constitute an intelligent interactive learning environment. The topic of the knowledge base may vary freely. With any topic, the system allows a student to interact with the predefined concepts and with those already defined by the student.

The ESBIS approach to computer based instruction does not assume that students understand everything ESBIS tells them, and it allows doubting students to request an explanation.

The explanations are managed with a tree data structure. Since a student may know most of the concepts involved in a "how" query, in the explanation, the program does not have to show every branch in detail. Thus, ESBIS explains only the first level when asked "how." However, if a student fails to understand any part of the explanation, the student can ask about more branches. In this way a student can access explanations to any level of the tree structure, depending on how much the student already knows. Thus, ESBIS well serves users with very difference competence levels.

On many occasions, the ESBIS explanations can throw light on what a student did not know and on where a student's concepts err. Besides providing a particularly good learning environment, the system supports experimentation with logic, a feature rarely found in most conventional CBI systems.

ESBIS does not automate teaching. It does not threaten to replace a human instructor but serves as the instructor's tool. It enhances students' learning and reduces the work of the teacher.

References

Clark, K.L., and McCabe, F.G.; micro-PROLOG: Programming in Logic. Prentice-Hall International. (1984)

Clocksin, W.F., and Mellis, C.S.; Programming in Prolog. Springer-Verlag. (1981)

Lee, M.C.; "Intelligent Tutoring System." Proceedings of the National Educational Computing Conference, San Diego, California, 116-120. (1986)

COMPUTER-BASED INTERPRETATION OF STUDENTS' STATISTICAL ERRORS: A PRELIMINARY EMPIRICAL ANALYSIS OF GIDE

Marc M. Sebrechts
Lael J. Schooler
Lafe LaClaire
Cognitive Science Program & Department of Psychology
Wesleyan University, Middletown, CT

Elliot Soloway
Department of Computer Science
Yale University, New Haven, CT

Abstract

Gide is an "intention-based" diagnostic system that analyses answers to statistical problems. This paper describes a preliminary evaluation of that system's effectiveness in examining student solutions and in diagnosing their errors. It provides a description of both the kinds of solutions that students generate and the strategies that GIDE must use to analyse those solutions. The data indicate that GIDE is capable of interpreting over 90% of the student solutions on the specific statistical test that has been analysed (a repeated-measures t-test). It is suggested that this approach can be used to design a computer aid for teaching statistics.

An Intention-Based Diagnostic Framework

There is substantial evidence to suggest that learning can be improved by providing students with the opportunity to explore a particular domain (Bruner, 1966). An important part of such computer-based "exploratory learning" environments is the ability to diagnose errors that a student makes along the way (Brown, 1983, Burton, 1982). This paper provides an empirical investigation of one diagnostic technique used to evaluate student errors in statistics, "intention-based" diagnosis. We will first briefly describe the theoretical argument for our approach to diagnosis. We will then describe the key elements of our computer-based diagnostic system for statistics, called GIDE, together with our initial empirical evaluation of the system.

Our strategy for diagnosis is predicated on the concept that analysis depends upon understanding what a student was trying to do. This framework is called "intention-based" diagnosis following the concepts implemented by Johnson & Soloway (1985; Johnson, 1985) for diagnosing semantic programming errors in Pascal.

This approach analyzes a solution by trying to describe how a student solution matches (or how it fails to match) an intended solution. It uses analysis-by-synthesis; it attempts to analyze a solution by building a plausible description of how that solution satisfies the goals defined by the problem.

For each problem, GIDE has a series of goals that must be satisfied by any correct solution. GIDE attempts to discover some way in which the student solution satisfies these requisite goals, by testing a series of alternative correct plans. If no such solution is found, GIDE attempts to discover the nature of the discrepancy. This is done through a series of erroneous or "buggy" plans that represent common conceptual errors, and "buggy" rules representing common mistakes across different types of plans. Details of the specific way in which the system works have been presented elsewhere (Sebrechts, LaClaire, Schooler, & Soloway, 1986; Sebrechts, Schooler, LaClaire, 1986); here we will concentrate on the demonstrated empirical import of several characteristics.

Many current systems attempt to provide diagnosis within a constrained problem context. The student is expected to first learn the way in which the system accepts solutions, and their solutions are evaluated accordingly. In our research, we are concerned with student solutions as they normally occur. There are two central questions that we address here: (1) Can the intention-based diagnostic framework be used for relatively unconstrained solutions? (2) What types of analyses are important for the domain of statistics?

Solution Analysis: An Example

Our preliminary analyses of GIDE examined performance on a repeated-measures t-test. This type of test can be used to determine changes in performance for a given group of individuals. The following illustrates this type of problem:

> Suppose that four classes of students showed mean performance of 64, 65, 77, and 62 on a standardized reading and mathematics abilities test. The students then participated in a course on problem-solving. Following the course, the mean performance on a similar standardized test for the four classes was 69, 77, 83, and 75. The mean changes (initial minus final mean scores) were -5, -12, -6, and -13. Is this change in performance statistically significant?

Examples of the types of solutions students generate for this test are shown in Figures 1 and 2: these are composites based on actual student solutions. The two solutions look very much the same, and an instructor might assume that the proposed solutions were quite similar. However, a more careful examination suggests that the superficially similar solutions reflect rather different conceptualizations. Student Solution 1 correctly computes the mean deviation on line 8. However, the student then incorrectly calculates the deviation scores, suggesting that the individual deviations are being added to the mean deviation rather than being subtracted. In contrast, in Solution 2, the student makes an error of dropping the negative sign in computing the mean deviation on line 8. Having done this, however, the remainder of the problem is correct.

GIDE's actual analysis indicated as the "bug report" after each problem reflects these differences. By using a goal and plan based structure, GIDE attempts to offer evaluation of the reasons underlying errors. In order to examine the range of errors and the feasibility of our approach an analysis was conducted of a number of student solutions.

Method of Evaluation

Subjects were 43 undergraduates who were taking a course in statistics. The subjects were generally unfamiliar with the procedure which we tested.

Each student was given instructions on how to perform a repeated-measures t-test and was asked to follow the instructions and provide a written solution to a problem while working independently in class. (Variations in the instructions across subjects will be described elsewhere. The key factor for the data reported here is that all students had instructions that were sufficient to solve the problem.)

Students wrote their solutions on paper, these solutions were then converted as precisely as possible into machine-readable format, taking the information from left to right and top to bottom. Line numbers were added by the system to facilitate reference to errors in the explanation. These programs were then submitted to GIDE for analysis.

Overall Performance of GIDE

Our first analysis considers whether or not GIDE was successful at analyzing the student solutions. A gross measure of success is provided by the number of lines in the student solutions (equations or expressions) for which GIDE can perform a correct analysis (i.e., can properly attribute the line as satisfying some goal).

The forty-three student solutions consisted of 1203 lines. Of these GIDE matched 971 fully and 16 partially. Thus, the system was able to "understand" 82% of the lines in the solutions. This performance appears roughly comparable to a teaching assistant for the course. There are 216 lines that are unmatched. However, 116 of these lines are "spurious" in the sense that a person knowledgeable in statistics could also not interpret

Student Solution 1

```
 1:   -5
 2:   -12
 3:   -6
 4:   -13
 5:   n
 6:   4
 7:   -6 + -13 + -5 + -12 = -36
 8:   dbar = -36/4
 9:   d_dbar
10:   -14
11:   -21
12:   -15
13:   -22
14:   d dbar2
15:   196
16:   441
17:   225
18:   484
19:   s2 = 1346/3 = 448.67
20:   s = sqrt s2 = 21.2
21:   sdbar = s / sqrt n = 10.5
22:   t = -9/10.5
```

*******Bug Report*******

Your calculation of the deviations was incorrect. A useful way to avoid this mistake is to note that the sum of the deviations should always equal 0. The sum of your calculated deviations equals -72.0

In the process of calculating the standard deviation, you added the individual deviation scores to the mean of the deviations; instead, you should have subtracted the mean of the deviaitons from the individual deviation scores. For example, where you have -14, you should have 4.0.

*******End of Report*******

Figure 1: A composite of student solutions and GIDE's associated comments.

Student Solution 2

```
 1:   -5
 2:    12
 3:    6
 4:    13
 5:   n
 6:   4
 7:   -6 +13 + -5 + -12    -36
 8:   dbar = 36/4
 9:   d_dbar
10:   -14
11:   -21
12:   -15
13:   -22
14:   d_dbar2
15:   196
16:   441
17:   225
18:   484
19:   s2 = 1346/3 = 448.67
20:   s = sqrt s2 = 21.2
21:   sdbar = s / sqrt n = 10.5
22:   t = 9/10.5
```

*******Bug Report*******

On line 8 you reversed signs.

*******End of Report*******

Figure 2: A variation of student solution 1 and GIDE's associated comments.

them. One-hundred meaningful lines were not matched and 1 line was misinterpreted. Thus, in gross terms, GIDE failed to explain about 8% of the lines.

Since student solutions were available during GIDE's development, the success rate we have described is not truly independent of the solutions. As an initial verification of these results, we analyzed an additional twelve solutions to the same problem which had not been available during the current development phase. GIDE successfully analyzed 96% of the lines for these solutions, indicating a success rate comparable to the first group.

In brief, this suggests that GIDE does a rather good job of analysis in this constrained context. Our main interest, however, is in understanding the types of analyses that are important for evaluation. The remainder of this paper will focus on the types of strategies that were needed for diagnosis.

Explicit and Implicit Goals

GIDE's analysis is goal-driven; hence the success of diagnosis depends on the extent to which student goals can be captured. One of the important characteristics of student solutions is that they can satisfy goals in two distinct ways: (1) They can explicitly state the procedure that satisfies a goal. (2) They can include statements that by reference indicate the implicit satisfaction of a goal that is not covered explicitly. In this case, a goal that is explicitly satisfied may depend upon the satisfaction of another goal that is not explicitly satisfied. By examining such goal dependencies, GIDE can infer what goals should be matched implicitly.

For example, in Solution 1, the variance is formulated explicitly on line 20: "s2 = 1346/3 = 448.67." This statement is interpreted as an explicit satisfaction of the goal of determining the variance. It consists of a sum-of-squares divided by the degrees of freedom (i.e. the number of subjects minus 1). The subject, however, never explicitly included the steps to compute the sum-of-squares. But variance depends on sum-of-squares; so the student must have determined sum-of-squares in some way, and it is therefore considered to be "implicitly" matched.

The presence of both types of goal matching is important to handle student solutions. In the set of solutions collected, there were 410 explicit goals and 35 implicit goals that we recognized. (The number of goals is substantially less than the number of lines since satisfying a goal may require several lines.) Of these 445 goals, GIDE matched 443. Most of these consisted of correct matches (approximately 98%). GIDE provided correct, explicit matches for 403 of the 410 explicit goals; it incorrectly matched 2 goals explicitly. In 5 cases GIDE provided implicit matches for explicit goals. In these cases, GIDE did not recognize the actual goal, but accounted for it through goal dependencies. Implicit matching seems to be necessary for the diagnosis of exploratory solutions which have loosely constrained syntax. In addition, by dependency relations, it provides a means to determine if a student has completed a goal that cannot be explicitly captured. In this case GIDE provided implicit matches for each of the 35 implicit goals. At the same time, however, this approach may be too powerful. In our sample, GIDE erroneously considered 8 goals to be implicitly matched that should not have been. This is a less than 2% false alarm rate, but more extensive study will be needed to test the limits of this diagnostic approach.

General Matching Strategies

Implicit matching is a single technique that is called whenever explicit matching has failed and a dependency link is active. Explicit matching, is a substantially more complicated procedure which needs to be sensitive to a wide range of student responses. In this section, we will describe the various kinds of matching which GIDE has available and offer a frequency analysis of their occurence.

As indicated above, there were 405 cases of explicit matching, 403 of which actually correctly matched the student solution. There are two general strategies that GIDE needs in order to analyze students' responses. First, it needs to be able to match any legitimate equations, both by their numerical value and by their form. In our data, there were 214 instances of equations, including frequent combinations of numerical and symbolic values. They do not follow any strong syntactic constraints similar to those required in programming; assignment to a value, for example, is frequent (see Solution 1, line 7).

The second general strategy consists of matching a free expression. The student may, for example, write down the value 4 to indicate the number of subjects, or use 36/n to indicate the mean. A teaching assistant who understands the problem can infer the referents of these expressions. Likewise, by using its plan structure, GIDE can identify such free standing expressions. In our data, there were 38 instances of this type of plan matching.

It is important to note that GIDE has the ability to perform matching in a stepwise fashion. It first looks for the appropriate general form of a solution, but will also accept equivalent forms. Thus, for example, on line 19 of Solution 1, the student writes "s2 = 1346/3." The general plan for variance expects a denominator of "n-1." Given that "n" is known to be 4 in this case, GIDE will accept 3 as a simplified form.

Group Variables

These strategies work well in accounting for single-line statements. In a number of cases, however, a goal consists of a group of elements. For example, a set of numbers to be summed or a set of deviation scores can be matched in any order. Alternative matching strategies are required to accommodate these sets which we term "group variables." In keeping with an "exploratory" problem structure, this approach allows students to list items in any order, without having to focus on a syntactic ordering constraint.

The most common use of group variables consisted of a list of numbers; lines 10-13 and 15-18 in Solution 1 are such lists. There were 106 instances of this type of group variable plan detected by GIDE. In some cases, students are actually explicit about the assignment of values to a variable name (e.g., dev1 = 13 - (-4), etc.), but in our study there were only 2 such instances.

Common Names

Legitimate equations, expressions, and lists should be analyzed by the plans described above. In some cases, however, none of these plans match the student solution. When this occurs, GIDE attempts to determine if there is an identifier in the student's solution that can be used as a clue to what the student was doing. It searches the proposed solution for a name that is associated with a current active goal and is recognized within the problem context. In Solution 2, for example, the average ("dbar" on line 8) is located by the commonly used name for the average "dbar." As with the more general matching strategies, the name may be part of an equation, a free expression, or a "group." In our sample, this type of "common name" plan was invoked 45 times.

Detecting Errors

So far, we have concentrated on the various techniques that GIDE requires to match student solutions. Matching, however, can occur in a number of different ways. In the simplest case, a match indicates that the student has correctly solved the problem; and in fact, most of GIDE's goal matches consisted of finding a correct plan for satisfying the goal. A match, however, does not necessarily imply that the goal has been correctly satisfied; it merely means that GIDE has accounted for the goal under analysis either by identifying a correct solution or by explaining the errors. In our sample, there were 96 cases in which matching a goal consisted of accounting for an error in the realization of that goal.

The basic strategy employed in GIDE is to construct one of several alternative accounts for how the student has attempted to solve the goal. This includes alternative "buggy" plans that consist of frequent conceptual errors, and "buggy rules" that are general errors outside the context of a specific plan. GIDE's analysis combines these plans and rules with the matching strategies described above. For the sample we studied, GIDE employed five categories of goal satisfaction: missing goals, plan errors, incidental errors, computational errors, and switched symbols. These categories are important as representative of the kinds of matching information that must be available to a diagnostic system.

If GIDE cannot locate a goal explicitly, either correct or buggy, GIDE waits to determine if it can be matched implicitly through some goal dependencies. If at the conclusion of its analysis, a goal is neither matched explicitly nor implicitly, it is considered to be missing. The goal is considered "satisfied" only insofar as GIDE has determined its status. There were 20 such missing goals.

In some cases, GIDE can determine that the student was using a defective plan to satisfy a goal. These are plan errors. In order to analyze such plans, GIDE must have available a list of potential plan deviations. For example, the normal way to calculate deviation scores is to find the difference between each deviation and the mean deviation. In our student solutions, a common plan error was the adding of the mean deviation to the individual deviation scores. For example, the first deviation score in Solution 1 (line 10) should have been -5 (a deviation score) minus -9 (the computed mean deviation). This would have yielded a value of 4. Instead the student wrote -14. Having failed to find the correct deviation, GIDE attempted to determine if the student had followed an incorrect plan and thereby identified the -14 as an incorrectly formulated deviation score. In this set of solutions, there were 6 cases in which GIDE indicated that a goal could be satisfied by this type of defective, "buggy" plan.

One way in which GIDE can track student solutions is by computation. Thus GIDE recognizes that "ave = sum / n" is an appropriate form for computing an average. Likewise, it can recognize "-9 = -36 / n" as a computational version of the solution. This procedure is complicated by the fact that students often make computational errors. In our current sample, computational errors accounted for the largest number of student errors (51 of 96). When GIDE detects a computational error, it notes the error and continues analysis of the problem. If the student makes an assignment using the erroneous value, GIDE keeps track of that error in its subsequent analysis. In this way it can separate certain types of conceptual errors from the effects of computational errors.

GIDE also needs to account for a range of incidental errors, such as leaving off the sign of a value (e.g., line 8, Solution 2). As noted earlier, even a small error can lead to dramatic differences in the interpretation of seemingly similar solutions (cf. Figures 1 and 2). In the present sample there were 4 instances of such errors.

Finally, the system needs to be responsive to inconsistencies in the solutions. Students occasionally identify the value of a symbol in the problem, but subsequently change that value. In our sample, there were 15 cases in which GIDE needed to recognize that a symbol had been switched in order to account for a goal satisfaction.

Summary

Our initial empirical study of students' problem-solving in statistics suggests that a wide range of diagnostic strategies are necessary in order to be able to account for a student solution. It is possible to incorporate a large number of these diagnostic techniques in a computer system using intention-based diagnosis. GIDE's success rate on these problems is comparable to that of many teaching assistants.

Although this performance is encouraging, there are several limitations to the data we have presented. First, students were provided with instructions for the problem solution. As a consequence, many potential goal and plan-based errors were avoided. In a less constrained context we would expect errors that are more varied, and we are currently testing such solutions. Second, we have tested only a single kind of statistical test, the repeated-measures t-test. Although this is useful in itself since this is an important statistical test, the system will need to be expanded to include a wider range of statistical procedures in order to be useful as a general diagnostic aid for statistics.

References

1. Brown, J.S. (1983). Learning by doing revisited for electronic learning environments. In M.A. White (Ed.). **The future of electronic learning.** Hillsdale, NJ: Lawrence Erlbaum Associates.

2. Bruner, J.S. (1966). **Toward a theory of instruction**. Cambridge, MA: Belknap Press of Harvard University Press.

3. Burton, R.R. (1982). Diagnosing bugs in a simple procedural skill. In D. Sleeman and J.S. Brown (Eds.). **Intelligent tutoring systems**. New York: Academic Press.

4. Johnson, W.L. (1985). **Intention-based diagnosis of errors in novice programs**. Research Report #395. Department of Computer Science, Yale University, New Haven, CT.

5. Johnson, W.L. & Soloway, E. (1985). PROUST: An automatic debugger for Pascal programs. **Byte**, *April*, 179-190.

6. Sebrechts, M.M., LaClaire, L., Schooler, L.J., & Soloway, E. (1986). Towards generalized intention-based diagnosis: GIDE. **Proceedings of the 7th National Ecuational Computing Conference**. San Diego, CA: National Eduational Computing Conference.

7. Sebrechts, M.M., Schooler, L.J., & LaClaire, L. (1986). Matching strategies for error diagnosis: A statistics tutoring aid. **Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics**. New York: IEEE.

COMPUTER SIMULATIONS IN ELEMENTARY STATISTICS

Florence S. Gordon
New York Institute of Technology
Old Westbury, NY 11568

ABSTRACT The present article will discuss the role of computer graphics simulations in the teaching of introductory statistics and probability. The author will focus on the use of a series of such programs developed and used in a variety of such courses and dealing with topics including random walk phenomena, the binomial distribution, the Central Limit Theorem, confidence intervals, linear regression analysis and the distribution of sample variances.

Introduction:

While most mathematics educators focus on the Calculus sequence and related courses, Statistics is fast becoming the single most important mathematics offering for the non-mathematics, science or engineering major. Several prominent groups concerned with mathematics education have called for statistical literacy to be an essential component of a modern college education. This reflects the way that statistics now pervades almost all facets of our lives.

Most of the tremendous growth in statistics and its applications (in fields such as business and industry, health sciences, economics, psychology, political science, educational research and so forth) over the last 30 years or so is directly attributable to the availability of high speed computers. As a consequence, we might expect to find a fairly extensive use of computers in statistics courses. Until very recently, however, this has not materialized to any great extent. This may be due to the almost universal adoption of hand-held calculators in such courses. It is only in the last year or two that we have seen a growing trend toward the use of computers by statistics students to perform statistical computations using prepared packages such as SPSS, Minitab, BMD, etc, especially in a main-frame computer environment. Simultaneously, there have recently been reports on a variety of efforts made to develop good software to address the educational needs of students in introductory statistics courses and this should lead to a major increase in the widespread use of computers in the variety of courses offered in statistics and probability.

Some Computer Graphics Simulations for Elementary Statistics:

The present article will discuss a number of programs developed by the author for simulating some of the underlying ideas in an introductory statistics course using computer graphics. They have proven very effective in terms of improving student understanding of the concepts as well as maintaining a high degree of student interest in the subject.

The typical elementary (non-calculus based) statistics course emphasizes the concepts and methods of inferential statistics rather than the computations. Similarly, even a solid calculus- based introductory statistics course also emphasizes the concepts and the theory over the computations. Despite this, the first generation of software for statistical education was computationally oriented and, as such, was rather inappropriate for these courses. At the elementary level, the result of using such software is usually little more than having the computer serve as a glorified hand-held calculator. This is certainly desirable in terms of eliminating most of the drudgery associated with doing the computations by hand. However, it is merely one small facet of the enormous potential for computer usage in statistical education.

Modern microcomputers, particularly with their graphics capabilities, provide an incredibly powerful tool to transform most of the statistical concepts and methods into visual representations which dramatically improve student comprehension. Most business- oriented software packages already provide the capability of displaying data in visual form via histograms, bar charts and pie charts and so such packages could be used in an elementary statistics course. However, this represents little more than the first week of the standard introductory statistics course.

Introductory statistics (both calculus and non-calculus based courses) can use the power of the computer for simulations and demonstrations of most of the important concepts and methods of inferential statistics. The range of possible uses of the computer is from very simple demonstrations up to complex simulations. The unifying motive should be to increase student understanding of the statistical concepts.

The one concept that pervades all of statistics is that of randomness. It is a particularly deceptive topic in the sense that few students truly grasp what randomness is all about. This problem is not always evident in typical classroom situations, but becomes painfully obvious if students are required to collect and analyze random data. They have no feel for what it means to make a random selection. The use of a table of random numbers seems little more than a totally unnecessary complication or nuisance to them.

The computer can provide a major assist in conveying some feeling for randomness in statistics. At a simplistic level, it can be used to generate repeated random selections from a set of pre-selected numbers, random selections of Heads and Tails to simulate coin tossing experiments, random generation of the results of dice rolling experiments and many other topics of a probabilistic nature. Through the use of computer graphics, students can achieve a much more effective visual demonstration of random phenomena.

Possibly the simplest and most accessible (to the student) application of randomness occurs in a two dimensional random or drunkard's walk problem. Such a program might draw the random path traversed for any choice of step size and number of such steps. Students are usually stunned to see the completely different paths that result from the same choices of step length and number of steps. (See Figure 1.) Their reaction clearly underscores their complete lack of comprehension of random phenomena. At an initial level, a program such as this gives the student a visual feel for randomness. In a more advanced course, the random walk phenomenon can be effectively tied to a variety of important and interesting processes.

In a related direction, the concept of randomness can also be approached effectively through the binomial distribution which can be simulated extremely effectively. In this program, the user will choose the number of trials N and the probability of success p and the program will randomly select sets of N repetitions of this binomial experiment, display the appropriate sets of successes graphically and keep track of the total number of successes that result during repeated trials. (See Figure 2.) Such a program gives the student a far better appreciation of what is happening than anything that can be done on the blackboard or with numerical displays alone. First of all, it gives a visual feel for randomness both in terms of the different results obtained on successive runs of the program for the same choice of N and p and also from an examination of the occasional outlier that arises. In addition, this program also gives a feel for the shape of the binomial distribution, and for the effects of different numbers of trials. Moreover, from the numerical summary of the results obtained (the mean and standard deviation of the number of successes actually obtained), the student sees a convincing numerical check on the accuracy of the theoretical predictions for mu and sigma in a binomial population. Thus, the students can first verify that the experimental results are "close" to those predicted based on probability and yet also see that the expected values are almost never precisely achieved in practice.

One of the major applications of the normal distribution in introductory statistics is its use to approximate the binomial distribution and this can be treated extremely well using computers. In particular, the binomial distribution can be displayed graphically as a histogram for any choice of probability of success p and number of trials N. The theoretical normal distribution can then be superimposed over the binomial histogram to provide a better understanding of why we can replace the binomial distribution with a corresponding normal distribution and for the need to introduce a continuity correction in such situations.

The Central Limit Theorem, while the primary mathematical tool of inferential statistics, is undeniably the least understood topic in any introductory statistics course. At the same time, it is the most obvious statistical topic for computer simulation and numerous people have developed such programs. In the author's version, the user has the choice of selecting any of various underlying populations and a desired sample size. The computer first draws the histogram of the chosen underlying population and then randomly generates the samples, calculates the mean of each sample, and graphs it. The resulting distribution of sample means literally grows in front of the user. Figure 3 shows the results of using the program with samples of size 9 drawn from a U-shaped population.

When this program is run several successive times using the same underlying population with successively larger values for the sample size, it is extremely simple to demonstrate the convergence of the sampling distribution to the normal distribution as sample size increases. Similarly, the graphic displays clearly show the effects of sample size on the standard deviations. If nothing else, a rough visual estimate of the spread of the sampling distribution can be compared to the width of the full screen to indicate the approximate values of one-half, one-third and so forth. Most students will even anticipate the formula for $\sigma_{\bar{x}} = \sigma / n$ from this visual analysis. Further, the program also displays numerical values for the mean and standard deviation of the set of actual samples selected. When these values are compared to the theoretical values predicted by the Central Limit Theorem for the distribution of sample means, it is extremely effective in validating the theory.

Estimation via confidence intervals is another standard topics in elementary statistics which students often end up treating mechanically with little comprehension. An appropriate program can do much to improve the level of student understanding using a graphic simulation to demonstrate the effects of different random samples on the confidence interval constructed. As shown in Figure 4, for a given confidence percentage, the program will generate approximately 75 different random samples of size n, for any desired n, and plot the confidence interval based on the mean and sample standard deviation of that sample. In this way, the students can see how different confidence intervals may contain or miss the true mean mu of the population as well as get a feel for the fact that a 90% confidence interval contains mu approximately 90% of the time.

Still another topic which lends itself to computer use is regression analysis. Numerous programs have been written to perform regression analyses numerically. In addition, a graphical display of the scatterplot and the regression line is also very effective, particularly in demonstrating the effects that changes in the data have on the resulting regression line. However, this topic also lends itself to graphical simulation. Typically, when covering this topic, the instructor seems to suggest that there is a single regression line developed for a set of data. However, the data on which the regression line is based is just one possible sample. Different samples will produce different regression lines. Thus, in the author's program, the student can select any desired sample size and the program will randomly generate repeated sample sets from an underlying population and graph the corresponding regression line. The result is shown in Figure 5. Based on this, the students can gain a better feel for the differences that can result from different samples as well as an understanding of the significance of the standard error of the estimate.

Finally, we discuss one other statistical topic that lends itself to a graphical simulation. While the Central Limit Theorem addresses the distribution of the sample means, it is also possible to consider the distribution of the sample variances. For example, it is known that when the underlying population is normal the distribution of sample variances has a chi-square distribution. This is clearly seen from Figure 6 based on repeated samples of size 8 drawn from a normal population. Moreover, for larger sample sizes, the resulting distribution is considerably smoother, more compact and more indicative of the chi-square distribution. For non-normal underlying populations, the resulting distribution is fairly obviously not chi-square.

Conclusions:

In summary, the computer now provides the opportunity for an instructor of elementary statistics to change the emphasis away from statistics as being simply a collection of standardized methods that the student must learn to apply mechanically. Instead, he or she can focus more on an understanding of the underlying statistical concepts. Hopefully, this will result in a more lasting appreciation of statistics on the part of the students and will go a long way towards achieving the goal of universal statistical literacy among college graduates.

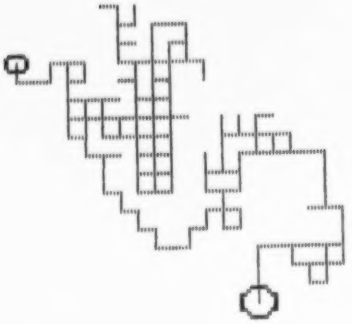


FIGURE 1

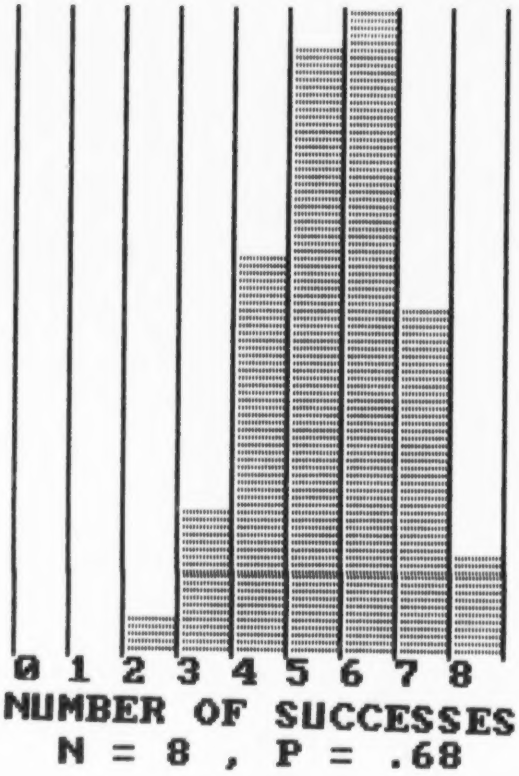


FIGURE 2

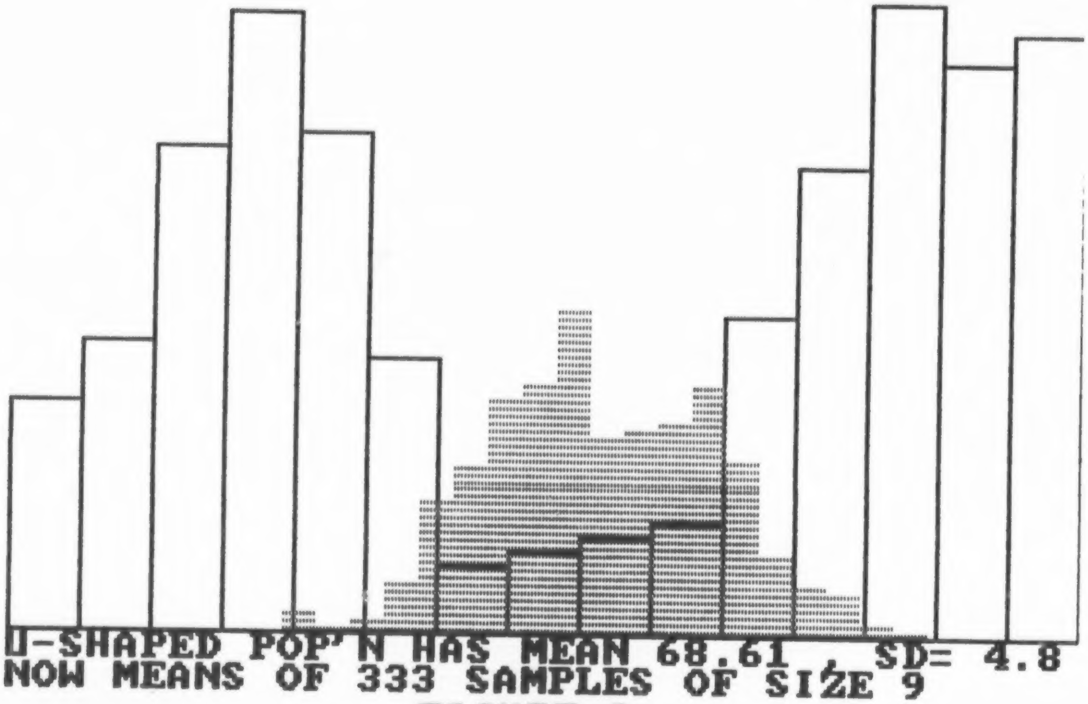


FIGURE 3

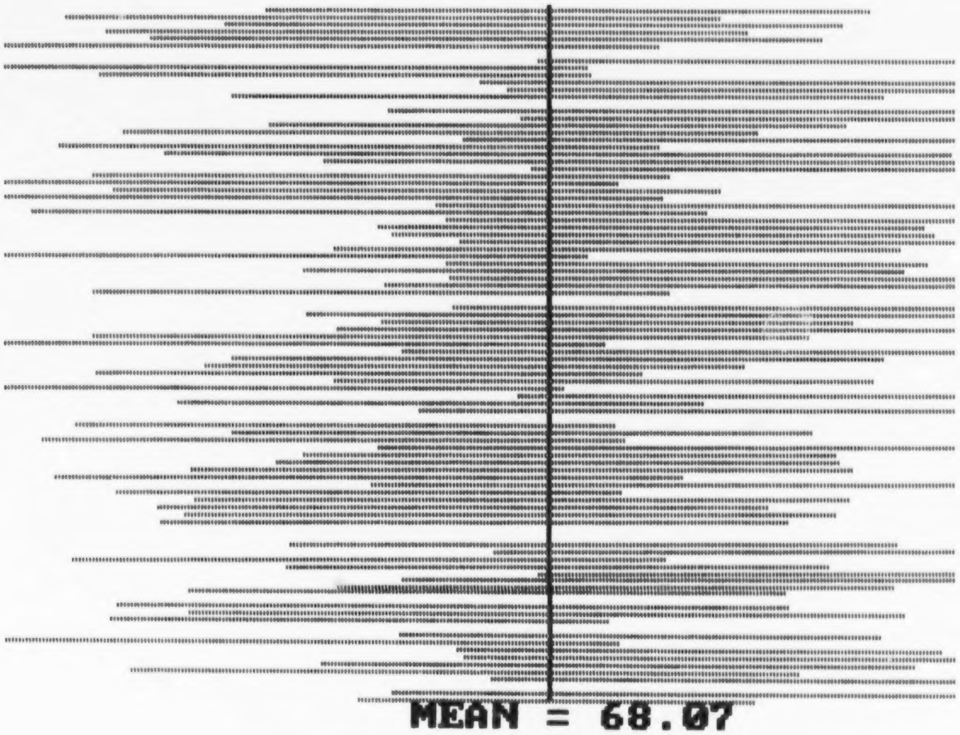


FIGURE 4

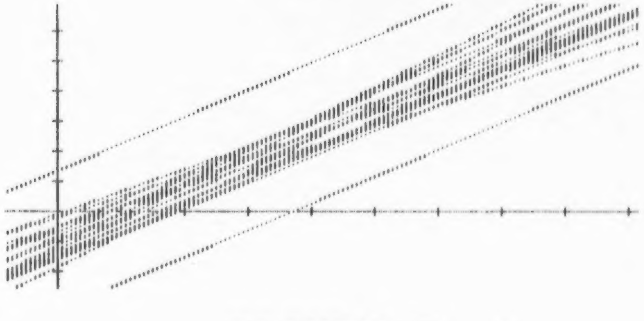


FIGURE 5

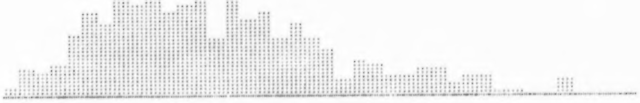


FIGURE 6

TOWARD LARGE-SCALE USE OF COMPUTERS

Alfred Bork, Chair
Information and Computer Science Department
University of California, Irvine
Irvine, CA

ABSTRACT

So far, computers play a minor role at all levels of our educational systems. But if we are to achieve the potential of the computer and related technology in aiding the learning process, we must investigate the possibilities of more extensive use. This investigation should take place in the context of the serious problems currently facing education. Advanced planning is essential; current directions are inadequate to our needs.

This session examines the factors that can lead to large-scale use of computers in education, and sketches some possible future directions. Both visions of future educational systems and plans to achieve them will be discussed. About half the session will be devoted to brief talks, and the other half to audience participation.

PANELISTS

Arthur Melmed
New York University
New York, NY

Paul Evans
IBM National Marketing Division

Alfred Bork
University of California, Irvine
Irvine, CA

Francis Dummer Fisher
Consultant

Sylvia Charp
T.H.E. Journal
Santa Ana, CA

BRIDGING THE GAP: STRATEGIES FOR PROMOTING THE TRANSFER OF LEARNING FROM COMPUTERS TO CLASSROOMS

Lynne Anderson-Inman, Ph.D.
University of Oregon
Eugene, OR

ABSTRACT

Integrating computers into the school curriculum requires more of teachers than a flair for the technical and a desire to be up-to-date. Serious attention must be given to the precise role of the computer as an instructional aid, the compatibility of software with existing curriculum, and the relative efficacy of computer based instruction over more traditional forms of promoting learning. The purpose of this presentation is to provide an indepth look at some of the curriculum and instructional variables that affect the integration of computers into the curriculum. Specifically, variables influencing the extent to which instruction or practice on the computer will transfer to the classroom (i.e., affect student performance on tasks within the existing curriculum) are examined and explained.

The transfer of learning from computers to classrooms is a critical issue for educators interested in maximizing the extent to which computer technology has a positive impact on student learning and performance. For example, the goal of most computer assisted instruction is improved performance on classroom based academic tasks, not computer based tasks. Use of computers in education assumes that this transfer of learning occurs automatically, i.e., that practice on the computer will result in a measurable performance improvement on tasks **off** the computer. Unfortunately, research and experience indicate that such transfer does not occur automatically.

The extent to which transfer of learning from computers to classrooms actually occurs seems to be influenced by many variables. In this presentation two major factors governing the degree to which computer based instruction impacts on classroom performance are explored: **curriculum compatibility** and **stimulus similarity**. Curriculum compatibility refers to the match between the software's instructional objectives and those found within the existing curriculum. Stimulus similarity refers to the degree of similarity across instructional formats and task demands. Using an array of visual examples from popular software and existing curriculum materials, the presentation will illustrate important dimensions related to these two factors and describe their potential impact on student performance.

Following this illustration, recommended strategies for promoting the transfer of learning from computers to noncomputerized classroom situations will be shared. These strategies have been found to maximize the extent to which computer based learning influences classroom performance and are presented as the beginnings of a technology for bridging the gap between computers and classrooms. Examples of how these strategies have been incorporated into both computer based and classroom based activities are included to illustrate the variety of possible transfer techniques. In conclusion, a four-step procedure for evaluating the extent to which a given software program will integrate into a teacher's existing curriculum will be shared, and evaluation forms for conducting this type of examination will be provided to all participants.

COMPUTHINK: PROMOTING COGNITIVE SKILLS

Beverly Saylor
South San Francisco Unified School District
San Francisco, CA

ABSTRACT

Participants will explore how to develop critical thinking and problem solving skills with the aid of computer and video programs. The presenters will provide:

1. A review of the CompuTHINK model and taxonomy of thinking skills and processes;
2. An explanation of how to use the CompuTHINK resource guide to select existing technology that matches and teaches thinking skills;
3. An overview on planning for and teaching cognitive skills within the content areas; and
4. Ways to adopt or adapt CompuTHINK to various programs.

The session will also include a demonstration of ways to use CompuTHINK-identified programs to develop selected critical thinking and problem solving skills. CompuTHINK is designated by the California State Department of Education as an Exemplary Educational Technology Project, funded for statewide dissemination.

COMPUTERS IN THE SCHOOLS: THE IBM MODEL SCHOOLS PROJECT

Hugh F. Cline, Chair
Educational Testing Service
Princeton, NJ

ABSTRACT

Objectives

The objectives of this session are to:

- Review the objectives, training, and implementation of the IBM Model Schools Program;
- Present an assessment of the program at the end of:
  a. the first year of Phase I implementation,
  b. the first year of Phase II implementation, and
  c. two years after the formal project and funding had concluded;
- Present the findings of the research studies;
- Review the critical success factors, at each of these three levels, that promoted effective implementation at the school site; and
- List a set of recommendations that could be adopted by a district or school as they attempt to promote effective integration of the computer as a tool in the high school curriculum.

The Project

The IBM Model Schools project was an attempt to promote the use of a computer as a tool in the instructional program. It was an attempt to move the focus from learning programming skills to using the computer, across the subject areas, for word processing, data base management, electronic spreadsheet calculations, and telecommunications.

IBM introduced two programs:

1. In the spring of 1983, IBM initiated an $8 million computer education program in three states, with 12 teacher training institutes and 89 high schools representing a cross-section of the nation's schools. The donation to each school consisted of laboratories of 15 PCs and extensive sets of software and word processing, data base management, electronic spreadsheets, tutorials, learning games, and programming languages. In addition, schools were provided with telecommunication equipment that permitted students, teachers, and teacher trainers to communicate by electronic mail. This program made possible the most extensive use to date of microcomputers, telecommunications, and commercially available applications software.

2. The second program began in the spring of 1984 when IBM launched a $12 million program in the 25 largest school districts in the United States. All schools and agencies received hardware and software of comparable value to the donation in the first program.

Thus, IBM has conducted two major computer education programs, which have attracted national and international attention. But how effective were these programs? Did they further develop the use of the computer as a tool in the high school curriculum? What was the impact of these programs two years after funding ended? Are there critical success factors that other schools and districts can use to promote the integration of computers into the school curriculum?

PANELISTS

Randy E. Bennett
Educational Testing Service
Princeton, NJ

Brian Stecher
Educational Testing Service
Los Angeles, CA

John F. McManus
Pepperdine University
Los Angeles, CA

Molly Watt
Educational Alternatives
Antrim, NH

Terence R. Cannings
Pepperdine University
Los Angeles, CA

Daniel H. Watt
Educational Alternatives
Antrim, NH

INTEGRATING MODEL-BUILDING AND SIMULATION INTO THE HIGH SCHOOL CURRICULUM

Nancy Roberts, *Chair*
Lesley College
Cambridge, MA

ABSTRACT

Model-building and simulation are useful approaches for the study of complex, interdependent problems. Such problems are found in the natural and social sciences, often crossing traditional discipline lines. Since most computer models are composed of equations, mathematics is an important part of model-building. System dynamics provides a set of conceptual and theory developing strategies that allow students to identify and study behavior over time. We believe high school students can measurably benefit from being introduced to this approach and these tools in multi-discipline areas. This project is developing curriculum modules that can be integrated into the natural and social sciences and mathematics.

To support the integration of simulation into the high school curriculum, a new software tool is being developed in addition to the curriculum modules. This tool combines data collected using probes, data entered by the keyboard, data displayed using a fast grapher or in table format, a rule procedure for entering formulas, and an iconic modeling program for building models of systems.

This panel reports on pilot studies where teachers integrated these materials into a variety of high school subjects. The panel consists of researchers, curriculum developers, and classroom teachers.

PANELISTS

Tim Barclay
Technical Education Research Centers (TERC)
Cambridge, MA

Jon Choate
Groton School
Groton, MA

David Clarkson
Brattleboro Union High School
Brattleboro, VT

Ellen Mandinach
Educational Testing Service
Princeton, NJ

Mary Budd Rowe
University of Florida
Gainesville, FL

SPONSOR: SCS

PRE-WRITING TOOLS FOR THE UNIVERSITY ENVIRONMENT

Nancy M. Ide, *Chair*
Vassar College
Poughkeepsie, NY

ABSTRACT

Writing tools for pre-writing tasks like planning, brainstorming, nutshelling, freewriting, organizing, and classifying are becoming increasingly available for the IBM PC and the Apple Macintosh. The first two pre-writing presentations discuss products under development as they are being used for research and instruction, and the following two presentations provide an overview and comparison of several existing instructional software products.

(1) *Pre-Writing: Visual Tools for Writing and Thinking*
Based on a cognitive model for written communication, a visual environment for writing and thinking developed at UNC-CH provides different modes of operation that correspond with the different kinds of thinking that go into writing. Experiments are providing information on the strategies writers (both novice and expert) use as they transform a loose network of ideas into a coherent integrated hierarchical structure and then write and edit the associated document. Experimental results will be presented.

(2) *Pre-Writing: UNC-WRITE*
UNC-WRITE, developed at UNC-CH, is being used in composition courses at the University of North Carolina at Chapel Hill to supplement classroom instruction. UNC-WRITE has helped students of varied backgrounds and abilities plan and organize their writing assignments. Evaluation procedures and findings, which have helped to determine how students learn to incorporate the program into their writing courses and assignments, will be presented.

(3) *Pre-Writing: A Comparison of Three Products*
The University of North Carolina at Chapel Hill has had the opportunity to evaluate three pre-writing tools currently available for the IBM PC: HBJ Writer, Writer's Helper, and Writing is Thinking. Pre-writing program capabilities have been described and critiqued by three English Department graduate teaching assistants who have taught beginning English Composition courses. Results of these comparisons will be presented.

(4) *Pre-Writing: Two Brainstorming Products*
Two products currently available for the Apple Macintosh permit unique approaches to brainstorming for pre-writing exercises. MORE is an outline processor that permits students to generate a hierarchical outline that can be rearranged as needed and can be viewed as single slides or bullets or as tree structures resembling organizational charts. Calliope is unique in that it permits the student to create separate units/ideas and then after the fact link them in the order that they should be presented. Both products will permit a student to manipulate and view the ordering of their ideas in many different progressions. Capabilities and comparisons will be presented.

PANELISTS

John B. Smith (1)
University of North Carolina
Chapel Hill, NC

Erika Lindemann (2)
University of North Carolina
Chapel Hill, NC

Margaret Hazen (3)
University of North Carolina
Chapel Hill, NC

George Brett (4)
University of North Carolina
Chapel Hill, NC

SPONSOR: ACH

WHAT ARE COLLEGES OF EDUCATION DOING ABOUT TEACHER TRAINING?

ABSTRACT

The four panel members represent four different universities that train a large number of teachers. Each has different ideas on the nature and extent of the computer-oriented training that preservice teachers should receive.

Dave Moursund
University of Oregon
Eugene, OR

The University of Oregon has offered both a master's degree and a doctorate in computers-in-education since the early 1970s. However, it has been slow to introduce preservice teachers to computers.

Recently, requirements for computer courses have been introduced into the teacher education curriculum. Depending on the program of study, these range from a one-credit course up to four credits (quarter hours). A variety of other courses are available as electives. In addition, some progress is occurring in integrating the computer as a tool into a number of courses preservice teachers take.

Other teacher training institutions in Oregon are making roughly the same progress in addressing the computer-related teacher training problem. However, there is no state-mandated requirement for this aspect of teacher training.

Dr. Charlotte Scherer
Bowling Green State University
Bowling Green, OH

The Clinical Experience and Computer Labs of the College of Education and Allied Professions at BGSU ensure that students have a number of computer experiences integrated into their overall curriculum. Students gain computer practice through workshops, tutorials, and simulations that help them learn various teaching skills, and through using productivity software to help them with their class assignments or lesson plan writing. Some also have limited experience with interactive video tape.

Also, elementary education students take a required "hands on" course, "Computer Utilization in the Classroom." Others are strongly urged to elect it. The students load and run programs, evaluate software, and discuss ways to use computers in various subject areas. They become familiar with productivity software and use authoring software that will enable them to create their own classroom computer resources.

Dr. Gary G. Bitter
Arizona State University
Tempe, AZ

The computer literacy requirement of the Professional Teacher Preparation (PTP) program at Arizona State University has the following catalog description:

> **CBE 300 Computers in Education.** An introduction to word processing, data bases, spreadsheets, teacher utility programs and evaluation of educational software. Required of all education majors (one semester hour credit).

Students are expected to demonstrate competency in the following areas:

1. The student will demonstrate an understanding of computers and their applications by identifying basic computer terms, using a microcomputer disk operating system, and other basic computer skills;
2. The student will learn to use a word processor, a data base, and a spreadsheet and cite educational applications for each;
3. The student will demonstrate an understanding of how to use teacher utilities including, but not limited to, graphic print programs, student worksheet generator, test generator, and a computerized gradebook;
4. The student will be able to name characteristics of educationally sound software and be able to apply these elements to identify software for the curriculum; and
5. The student will demonstrate an ability to design lesson plans applicable to software programs from their academic major.

Larry Hannah
California State University
Sacramento, CA

Recent California legislation concerned with teacher training in the area of computer education is having a profound impact on schools of education in the state. This legislation:

1. Requires that all teachers receiving a clear teaching

credential after July 1, 1988, must satisfactorily complete "computer education coursework, which includes general and specialized skills in the use of computers in educational settings";

2. Gave the Commission on Teacher Credentialing the authority to develop and disseminate "voluntary standards for the training and performance of teachers and resource personnel in the area of computer education;" and

3. Mandated that the Commission study the effectiveness of the training and performance of teachers and resource personnel in the area of computer education and report the results by December 31, 1987.

This presentation will outline guidelines developed by the Commission's Computer Education Panel and implementation efforts being made by individual schools of education and the California State Universities and Colleges system.

SPOTLIGHT ON BUSINESS AND INDUSTRY

Mehdi Khosrowpour, *Chair*
Pennsylvania State University
Middletown, PA

Harry Albright
Verl Dennis

ABSTRACT: Learning to Exploit the Information Center

Harry Albright, Pennsylvania Blue Shield, Camp Hill, PA

This presentation will highlight some key issues in the implementation and maintenance of an information center. Experiences from Blue Shield will be discussed.

ABSTRACT: Videodisc Use in Management Development: A Discovery Learning Case

Verl Dennis, Arthur Andersen & Company, St. Charles, IL

This presentation highlights various computer-based techniques in Arthur Andersen & Company's training programs. A specific example of videodisc use in a management development setting will be discussed.

COGNITIVE BARRIERS TO INITIAL PROGRAMMING INSTRUCTION: A SURVEY OF RESEARCH

William Wresch

Department of Mathematics and Computing
University of Wisconsin - Stevens Point
Stevens Point, WI 54481

ABSTRACT

There are many barriers to programming instruction. One is inadequate programming environments: poor editors, cumberson filers, nonexistent debuggers and tracers. A second problem is confusion over instructional environment: should student be given little instruction while they "discover" programming constructs, or should they be led through every possible piece of syntax? A third problem area is program design. Structured problem solving requires domain-specific knowledge, and leads to errors involving the combining of program segments. Of course the last major problem is program coding. Student misunderstanding to tne essential operations of computers can lead them to assume the computer has interpretive powers similar to those of people. Such misunderstanding can lead to three categories of coding errors.

What makes learning to program hard? In the last few years we have all seen a growing number of reports on the cognitive barriers to learning programming -- the barriers that prevent students from succeeding in even an initial course involving introductory programming instruction.

One way to group the reports investigating barriers to programming instruction is to categorize them into four main areas: programming environment, instructional environment, program planning, and program coding. Other groups might be just as useful or more complete, but these four areas cover most of the issues faced in classes involving initial programming.

Programming environment

Programming environment refers to the tools available for programmers. At its simplest level this would be the operating system used for program development, but environment could also include books and people resources available to novice programmers.

Programming environment has a dramatic effect on the efforts required of novice programmers. For instance, one of the major complaints often made about Applesoft BASIC is the lack of an editor. Since even the simplest insertions and deletions can involve complicated command sequences, it is often easier to retype a line than to edit it. So program development is unnecessarily clumsy. Actually this has nothing to do with BASIC as a language. It is a problem with the programming environment sold with that version of BASIC, and it is a problem that dramatically increases the effort required to write programs, and distracts novices from the task at hand -- learning to program a computer.

Other aids potentially available in a programming environment include debugging tools, error trapping routines, subroutine libraries, and file management routines. Such tools are routinely available to programming professionals, but are often lacking in courses taught to novices.

The differences in programming environments are so critical that they often dictate our choice of instructional language. For instance, Roy Pea points out that "Many of the alleged advantages of Logo over BASIC as a language for children actually lie in the sophistication of Logo's programming environment. (Pea 25) Some of the newer language implementations such as Turbo Prolog and Instant Pascal go beyond a good editor and clear error messages, to include a trace debugger that displays the program execution in one screen window and points to the current program line in another window. Such tools allow programmers to see their program as it executes step-by-step and greatly reduce the effort required to debug code. Such tools are so helpful they should be a factor when instructors choose language implementations.

But programming tools aren't the only aspects of the programming environment that help determine success for novice programmers. In a study of children who had developed significant expertise in programming, D. Midian Kurland found access to computers and access to experts were both common factors in their success. In every case, however, this access was outside of school, as these experts reported that "their schools did not have the equipment, books, manuals, or knowledgeable people required to help them with their programming problems" (Kurland 16).

The need for access to computers stemmed from the need to use computers for lengthy periods of time while developing expertise. Kurland quotes a computer professional who describes his own training:

> I was always struck at how much time I put into school compared to everybody else I knew. . . (Those) who were taking ordinary computer science curriculum compared to English, it was at least double the amount of time. People were just putting in anazing amounts of time. (p7)

The child experts Kurland interviewed mirrored this need for computer time. All of them owned their own computers and often spent additional hours at university computer centers. Easy and regular access to computers was a necessary part of their growth as programmers and a universal shortfall of their schools.

The other area where these experts had needs was in access to expertise. Two of them satisfied this need by hanging around university computer centers where they could get help from people at the centers. Another was tutored by an adult friend. All of them communicated to other programmers through electronic bulletin boards. By whatever means, they were all able to establish an environment that included ready access to experts.

Access to computers and access to experts can be crucial to novices as well. We don't expect novices to spend many hours using computers, but they shouldn't be rushed on and off a machine, either. As we attempt to establish computing environments conducive to learning, we should remember that such environments require more than good editors and sophisticated debuggers.

Instructional Environment

Discussions of programming environment lead naturally to discussions of instructional environment. How much should we teach, and how much should we let students discover on their own? Logo advocates used to be especially emphatic about letting students discover facts for themselves while performing activities that were meaningful to them. Teachers were to facilitate, but not to lead, to serve as a resource rather than as a director. The difficulties with this approach are well chronicled. Pea (p27-28) describes one such experiment in which students were taught basic Logo commands and then encouraged to develop their own projects. Teachers gave only minimal instruction in advanced Logo concepts, required no assignments and gave no tests. The result was little growth in programming ability. The teachers changed their approach dramatically after one such year.

While we all probably find such results unsurprising, we should be aware that the opposite extreme can be just as unproductive. Many of us have had the occasion to observe mind-numbing classes in which a new day meant a new BASIC instruction, and a new program meant another trivial exercise. An incredibly candid article recently described just how far such regimentation can be taken and how disasterous it can be.

John Dalbey and his colleagues at UC-Berkeley attempted to improve student planning strategies by teaching a class of junior high school students the use of structure templates. The point of such templates was to help students identify the fact that most programs use the same essential control structures over and over. Problems could be identified (and often are by experts) as "an accumulator loop" or "a case structure." Dalbey used very simple structures with these students. Students were to use these structures as an intermediate step between program specification and actually coding their program. The point was to help them think of the program as a collection of "chunks", chunks that were common from one program to another.

Students in the experiment were given five classes of instruction in use of these templates, but had trouble translating the use of templates to similar problems and failed to see the benefit of using the templates. Students preferred to be on-line as they developed their programs. At this point classroom atmosphere became so bad that "serious discipline problems arose in this class" (p16). Clearly there was a limit to how much structure students were willing to accept as they developed their programs.

Even without experiencing open rebellion, most of us have been aware of the effects of too much direction when teaching young students. Students need a structure for their work, but they still need to "own" what they are doing. A nice middle ground seems to have been developed by Kenneth Kwajewski who starts out his courses with solid instruction in the problem solving sequence, and then gives his students substantial programs to write. Most of these programs involve teamwork, and will ultimately be used by other students in the school (one of his programs involved creating an SAT preparation program). He continues to lecture to his students once every two or three weeks, but most time is spent with students actually working through the substantial programs he gives them. The fact that these programs will be used by fellow students not only adds motivation, but forces students to focus on user interfaces and error trapping, areas generally omitted in initial courses. The fact that programs tend to be huge forces students to use development teams and modular design, areas also usually absent from most programming courses. Kwajewski's approach shows that even in initial programming classes there is a choice other than pretty pictures or endless workbooks.

Assuming that we take some kind of instructional middle ground, leaving room for students to explore while not leaving them to wander aimlessly, we now face the question of how best to foster their growth. That takes us to the two remaining areas of research -- student use of plans, and student growth in program coding.

Program Planning

Dalbey's attempt to have students see a program as "chunks" is common to those who see top-down refinement as a primary planning tool. We intuit that a program would be easier to write if we did it in smaller pieces. We begin by taking a larger task and determining what its constituent tasks might be.

This attention to chunks is based on some widely reported studies of how expert programmers remember code. Numerous studies have indicated that experts group lines of code in their memory according to their function. In one study in particular, Adelson found that experts remembered lines of code in the order in which they might be run, while novices grouped lines by syntactic category. Since chunking seems to happen automatically with experts, why not make such an activity explicit for novices?

There seem to be two reasons why this process is harder than we would expect and both are tied to the question, what is a chunk? To begin with, in order to break a problem into its constituent parts, we have to fully understand that problem. Pea reminds us that if a child were asked to write a graphics program to draw a Colonial home, she would have to know what Colonial houses looked like, what their major elements were. With little understanding of such houses she might have a little sense for their components. Telling her to subdivide her task might be very little help -- she has no sense for appropriate subdivisions. We get an analogous situation when we assign mathematically oriented programming tasks. Some of our students will grasp the mathematics sufficiently to see the components of the problem. Others will be stuck trying to subdivide their confusion.

The second problem with chunks is their relation to programming ability. Is a chunk for a novice programmer the same size as a chunk for an expert? Eliot Soloway points out that while students are universally told to break a problem into parts, they are never told how many parts. Why _n_ levels and not _m_? His response is to break a problem into subproblems "on the basis of problems you have already solved and for which you have canned . . . solutions" (p855). Novices of course have very few canned solutions. Telling them to break a problem into its parts simply doesn't get the same results as telling this to an experienced programmer.

This all means that while we may want to introduce structured problem solving, we should be aware that this methodology requires domain-specific knowledge about the problem being programmed, and a growing repertoire of program routines.

Even if we can achieve this increased structure, it doesn't mean our problems are over. At least one study has indicated that over half of novice errors are due not to syntax confusion, but to mistakes involving higher level plans.

Soloway details some of the ways program components can be combined and how these combinations can lead to errors. To him, components can be combined through abutment, nesting, merging, and tailoring. Abutment involves placing components in sequence. Nesting occurs when one component is completely surrounded by another. In merging, two components are interleaved. Tailoring involves modifying a stock component. A study of novice university-level programmers by Spohrer and Soloway showed that students created most of their errors by miscombining smaller structures. For example, Spohrer describes a "summarization" problem as occuring when novices merged two components and impare one of the components in the process. Other cases of problems are optimization, previous-experience, specialization, natural-language, interpretation, boundary, cognitive load, and unexpected cases. All of them involve errors created by misapplying or miscombining program components.

Clearly structuring a program is not as easy as it might seem and hardly constitutes an automatic solution to the problems of novice programmers. We also understand that even if structuring a problem were simple, it still leaves the problem of coding each structure, a problem that brings a whole new set of errors with it. These errors have been studied in some depth and have been carefully grouped and labeled.

Program Coding

To begin with, there is solid evidence that the misunderstanding of program constructs is not just a problem for children, but is present for all novices whatever their age. Jeffries (1982) reports confusions over variable scope and iterative processes were common among college students even _after_ they had completed a Pascal class. If even trained college students have trouble with some of the basic concepts of programming, we can imagine the problems for novice high school students.

These problems in program coding have been described by several researchers. Sleeman and his colleagues at Stanford have described eight categories of typical novice errors: assignment statements, print statements, read statements, variables, loop construction, if statements, other flow of control problems, and debugging errors. As an example of READ errors, Sleeman gives this BASIC code.

```
10 READ LARGEST
20 DATA 2, 9
30 PRINT LARGEST
```

Students in his study often slected "9" as the output of this program segment, clearly misunderstanding how the code would actually operate.

At least two theories have been offered for why students make such errors. One theory could be called the "natural language theory," while the other might be called the "Super bug theory."

The first theory has been summarized by Bonar (1985) as follows: "the key idea is that many programming bugs can be explained as novices inappropriately using their knowledge of step-by-step procedural specifications in natural language." Bonar bases his ideas on interviews with novice programmers as they attempt to write programs in Pascal. In one case a student creates an REPEAT...UNTIL THEN construct, and justifies it in an interview: "It makes sense because you are repeating until that (points to UNTIL line) and THEN (his emphasis) what do you do? Well, THEN take the average" (p7).

In another example, a student misapplied the word "while." Since "while" means continuously in English, he assumed there would be a continuous check for a test value in a while loop. "Every time (the variable tested in the WHILE condition) is assigned a new value, the machine needs to test that value..." (p13). The student didn't understand that the While test condition would only be checked at the completion of the loop, because that's not what WHILE means in English.

Another common natural language problem occurs when novices first attempt compound IF statements. If a test is required to select a quantity that is greater than one amount and less than another, students generally begin by writing it this way:

```
IF (GPA > 2.5 AND < 3.2) THEN
```

Since that is how a compound comparison is stated in English, that is how they first construct it in a programming language. They don't see the need to repeat the variable in the second comparison, since we wouldn't express the comparison that way in conversation.

For Roy Pea, however, the root of most coding errors is not so much English, as it is the "Super Bug," the idea that "there is a hidden mind somewhere in the programming language that has intelligent, interpretive powers." (p9) This super bug leads to three specific types of errors novices make as they program.

The first is the "parallelism bug." If there is a hidden mind in the computer, then it can process several things at the same time as humans do. Pea uses the following BASIC sequence as an example:

```
IF SIZE = 10 THEN PRINT "HELLO"
FOR SIZE = 1 TO 10:  PRINT SIZE
NEXT SIZE
```

Novices who believe the computer can function as a person, will expect the output from the above sequence to be 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, HELLO. They would expect the computer to print HELLO anytime size was 10, even though the computer totally ignores the IF clause after it has passed it.

Pea's second category of bug is the "intentionality bug," or the idea that the computer program can set its own goals. He uses the following Logo code to demonstrate:

```
TO SHAPE :SIDE
IF :SIDE = 10 STOP
REPEAT 4 (FORWARD :SIDE RIGHT 90)
SHAPE :SIDE/2
END
```

While the intent of the above recursive procedure is to draw a series of boxes inside each other, some students predict the procedure will draw only one box with a side of ten. Why? Because even though :SIDE may start out to be 40, the program "knows or sees what will happen elsewhere in itself," and so it stops itself automatically when it sees the FUTURE value of :SIDE will be 10.

Pea's last category of super bug is "egocentrism bugs." In this category, students assume the computer can understand their meaning even if there is nothing in the program to make their meaning explicit. For example, novices may expect REPEAT 4 (FORWARD 30) to draw a box even though they have left out the turn command. They assume that somehow the program will understand what they mean and perform their intentions, rather than their instructions.

These instances of "super bug" coding errors help demonstrate why programming can be so difficult for novices, but they also show how programming instruction can be useful to students who may never be more than computer users. Writing programs forces them face to face with what computers really are. They are forced to appreciate the limitations of computers and the need for explicitness. Even if they never write a program in BASIC again after their six or eight weeks of instruction, they have gained a valuable lesson in computer "literacy." And since many of them will program again, even if those programs only take the form of spreadsheet macros, they will be better prepared to use the power of the computer.

We understand that teaching novices even elementary programming is not easy, but with appropriate programming and instructional environments, and with a better sense of useful problem-solving techniques, computer programming instruction is possible. In fact most of the research into the difficulties novices face when they learn to program demonstrates the importance as well as the difficulties of initial programming. What better way to teach students the essential processes of the machine?

Bibliography

Bonar, J.; Soloway,E. Preprogramming knowledge: a major source of misconceptions in novice programmers. ED 258805, 1985.

Dalbey, J.; Tourniaire, F.; Linn, Marcia. Making programming instruction cognitively demanding: an intervention study. ED 257441, 1984.

Jeffries, R. et.al. The processes involved in designing software. In J.R. Anderson (Ed.) Cognitive Skills and their acquisition. Hillsdale, NJ: Erlbaum, 1981.

Kwajewski, K. Teaching BASIC through programming, not lecture. The Computing Teacher, 14, October 1986, 7-11.

Kurland, D.M.; Mawby, R.; Cahir, N. The development of programming expertise in adults and children. ED 257441, 1984.

Luehrman, A. Spreadsheets: more than just finance. The Computing Teacher, 13, April 1986, 24-28.

Pea, R.D.; Kurland, D.M. On the cognitive prerequisites of learning computer programming. Technical Report No. 18, New York: Bank Street College of Education.

Pea, R.D. Language-independent conceptual bugs in novice programming. Technical Report No. 31, New York: Bank Street College of Education, 1984.

Sleeman, D.; Gong, B. From interviews to policy recommendations: a case study in high school computer programming. ED 257415, 1985.

Soloway, E. Learning to program = learning to construct mechanisms and explanations. Communications of the ACM, 29, September 1986, 850-858.

Spohrer, J.C.; Soloway, E. Novice mistakes: are the folk wisdom's correct? Communications of the ACM, 29, July, 1986, 624-632.

THE GRAPHICAL REPRESENTATION OF PROGRAMMING: RESEARCH INTO ALTERNATIVES FOR LEARNERS

Nancy Cunniff
Robert P. Taylor
Center for Intelligent Tools in Education
Teachers College, Columbia University
New York, NY 10027

Abstract

This paper presents an argument for using the computer to provide learning alternatives. It focuses on graphic representation as an alternative to traditional textual representation of concepts and knowledge, and it presents a rationale for research and development work in that arena. The paper uses the teaching of programming as an illustration of an arena where alternatives can affect the learning of novices. It discusses FPL, a graphically represented programming language, and summarizes on-going research into the effect of graphic versus textual representation in student work involving that language and Pascal, a textually represented language.

Introduction

> *I soon felt that the forms of ordinary language were far too diffuse...I was not long in deciding that the most favorable path to pursue was to have recourse to the language of signs. It then became necessary to contrive a notation which ought, if possible, to be at once simple and expressive, easily understood at the commencement, and capable of being readily retained in the memory.*
>
> -**Charles Babbage**, *On a Method of Expressing by Signs the Action of Machinery*, 1826

The need for alternative ways to represent ideas, concepts and knowledge has long been acknowledged. Because different learners have differing capabilities for processing, understanding and remembering material, no single mode of presenting concepts works equally well for everybody. Alternative modes are essential. This is the central presupposition of this paper and the work it reports.

By contrast, for historical and other reasons, formal education seems to function upon a radically different presupposition: textual representation is adequate for nearly everything. The work reported here moves to redress this imbalance because it assumes both that it needs to be corrected, and that computers increasingly provide a means for doing so.

Following the intuition of Babbage and many others (Polya, 1957; Arnheim, 1969; Shneiderman, 1983; Lodding, 1983), this work concentrates on graphical representation as an alternative to textual representation. It involves empirical research because there is so little research on the effectiveness of graphical representation. The content area is the teaching and learning of programming because: (1) the work was centered in an environment where programming is taught and (2) significant parts of programming lend themselves to graphical representation. The implications, though, are general.

This paper proceeds by discussing

1. a rationale for provision and exploration of the graphic alternative,
2. a graphic alternative for the teaching and learning of programming,
3. a research challenge: what we are currently doing and what we think needs to be done.

Rationale for Studying the Graphic Alternative

Educational Issues

Why should educators be concerned with alternatives such as the graphic alternative? Two reasons are obvious. First, individual human beings differ considerably from each other with respect to their strengths and the way they learn (Gardner, 1983). For example, some seem to learn best from reading, some from seeing, some from hearing, and so on. Second, some topics may be more lucidly perceived by *all* learners when presented in one particular form rather than any other.

Nevertheless, provision of alternatives is too often ignored in the creating of teaching materials and curricular activities. Western education often proceeds as if the "written world" is sufficient for teaching about the real world (Huggins & Entwisle, 1974). A vast majority of students' learning time is spent manipulating and creating text. New materials and approaches all too often merely involve the use of *different, but still text-based materials.*

In consequence, successful students are, too often, those who have learned to manipulate text easily, rapidly and purposefully. Those who have trouble with text are, quite often, unsuccessful in our educational institutions. The system offers little support to those who aren't textually apt, ignoring or abandoning even those with highly developed alternative styles and aptitudes. Computing can and should be used to correct this, providing, of course that research confirms the intuitions regarding graphic representation.

Computing Issues

The impact of hardware and software capabilities. Technically, computer hardware has been capable of producing high quality graphic displays for quite some time. Recent advances in software development have begun to exploit this capability. However, despite rapid technical improvements and price decreases, costs remain relatively high. For educational institutions, especially, proof of the effectiveness of graphic representation must precede any commitment to a significant demand for such use and the concomitant allocation of funds.

Provision of new alternatives. Decisions about computer use in education should be driven by the search for alternatives: how can knowledge be represented *differently* so that it is more easily understood, applied, integrated? The very nature of computing allows us to represent dynamic and spatial relationships more easily than can be done either in text or with static concrete materials. However, with respect to both finances and time, it remains less costly to use computers merely to mimic familiar modes of knowledge representation. Even artificial intelligence researchers, who are seeking to identify and explore new relationships and knowledge structures, spend little time exploring how to represent these relationships in non-traditional non-textual ways (Dear, 1986).

Influence on schools. The continued appearance of educational software in the "electronic workbook" mode only demonstrates that the representational richness made possible by computer hardware and software development tools has yet to be tapped in the educational arena. Without solid research pointing out the advantages of the alternative approaches which the computer can support, the potential for computing to provide such alternatives may not be realized. Moreover, we may fail to take advantage of the potential positive impact which computing could have on learning. Whether education benefits from the powerful potential of computing will depend, to a large extent, on whether research can show that alternative uses are more effective than merely transferring traditional, textual representations onto the computer display.

Issues in Programming and Teaching Programming

Teaching programming. The realm of teaching and learning programming provides an ideal environment in which to investigate the effect of alternative presentation modes for several related reasons. First, in the world of computer programming, though experts have always relied on graphics of various sorts to clarify their arguments, programming languages have remained largely textual even though they incorporate ideas that are clearly representable graphically. This textual focus may be fine for students who are inclined to see things textually, but for students without highly developed linguistic intelligence (Gardner, 1983), such a textually-based approach is not particularly helpful. Consequently, students have been forced to study this abstract, conceptual subject matter in an environment based on symbolic representation. There have been no real alternatives for learners who may have difficulty dealing with either the abstract, conceptual nature of programming or the symbolic nature of text.

Second, there are a large number of students studying programming on many levels of schooling. Additionally, there are many students who might wish to study programming but who are kept from doing so by the overwhelmingly symbolic nature the subject matter and the limitations of available programming languages. We think that a graphic representation may provide a more concrete representation needed by some learners, thus helping them to grasp the abstract nature of programming. This would help those already studying programming and encourage those currently afraid to begin.

Finally, and maybe most importantly of all, we are still trying to figure out *how* to teach programming, what materials and methods to use, and what sequence of conceptual presentation to follow. Because we are at such an early stage in the development of educational processes and products for the teaching of programming, educational researchers can and should address these questions, and then propose sound, empirically verifiable approaches for the development of alternative approaches.

Visual programming. The idea of visual representation in programming is not new. Over the few decades during which programming has been taught, there has been increasing interest in the visual representation of algorithms and representation modes. Early attempts to define an alternative mode for representing logic in programming gave rise to standard flowcharts. They became a widely used, popular vehicle for planning and debugging program logic and algorithms. Despite ongoing debate over the usefulness of this particular tool (Brooks, 1975; Shneiderman, Mayer, McKay & Heller, 1977; Shneiderman, 1980), the value of graphic representation was recognized, and many more attempts have been made to design viable tools for graphically representing programming (see Schneyer, 1984; Myers, 1986).

Recent developments and rising interest in the topic have given rise to the genre of "visual programming," defined by Grafton and Ichikawa (1985) as encompassing three distinct areas of research:

1. "graphic techniques that provide both static and dynamic multidimensional views of software,"
2. "graphics-based very high-level programming languages," and

3. "animation of algorithms and software." (Grafton & Ichikawa, 1985, p. 7)

Most development has focused on the first and third of the areas enumerated above, producing systems for graphic display of the program flow and data values during *execution* and *debugging* (Raeder, 1985). Although this focus is important in the sense that it makes visual the dynamic nature of a program, it fails to address the need for graphically represented languages to be used in the actual planning and writing stages of programming. Development of graphics-based languages lags far behind development of programming support environments. Even in computing environments such as the *Macintosh*, renowned for its iconic interface, programming languages still rely almost exclusively on manipulation of text for the actual coding phase of the programming process. (For example, see the Macintosh Pascal software by Maruhnic, Lucas, Singer, Conway & Huras, 1984).

The growing interest in visual programming suggests that some developers, at least, believe graphic tools make it easier to understand the complex action of the computer. If these tools help experts to understand complex systems, it seems reasonable to assume that a visual representation of programming constructs and logic would help novices understand programming more easily and quickly. However, while intuition and anecdotally recorded observation may have convinced many that graphic systems are viable, empirical research verifying that viability is badly needed.

The Context: A Graphic Programming Language

For several years we have been teaching programming to novices using FPL, a graphically represented programming language under development at Teachers College, Columbia University. The testimonies that FPL is learnable, by a large number of novice programmers with little mathematics or science background, has reinforced the intuition that a graphical representation is useful. It has also suggested that we should seek empirical verification of this intuition. The language is fully described elsewhere (Taylor, 1982; Taylor, Cunniff & Uchiyama, 1986), so the following comments give only the briefest summary and present some idea of the appearance of the implementation of the language.

FPL (First Programming Language) is a graphical representation of classical programming designed for teaching programming to novices. In FPL, all programming constructs and concepts are represented graphically, using icons and spatial arrangement to represent the programming actions. FPL is built around a set of icons each of which stands for a specific programming instruction. There are eleven FPL icons; each represents a specific programming action. Eight of the icons ultimately include programmer inserted text, the variables and constants of the program. The icons replace the "reserved words," or instructions of text-based programming languages and thereby embody flow of control and logic. Figure 1 illustrates the FPL icons and, for comparison, their corresponding Pascal translations.

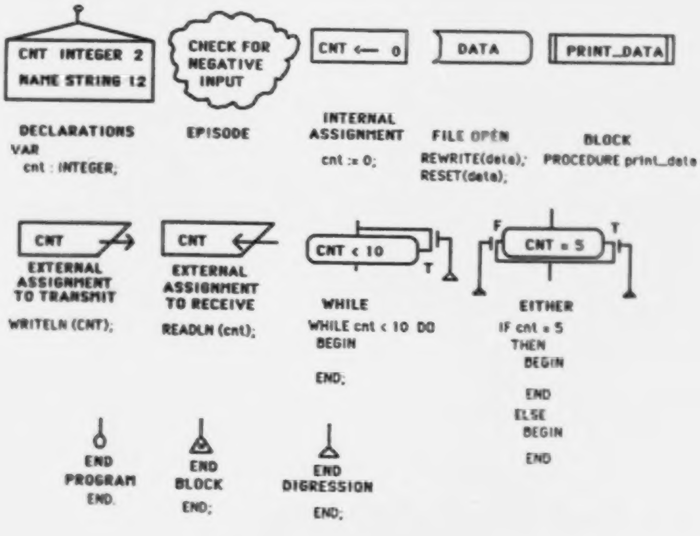

CNT INTEGER 2
NAME STRING 12
DECLARATIONS
VAR
cnt : INTEGER;
CHECK FOR NEGATIVE INPUT
EPISODE
CNT ← 0
INTERNAL ASSIGNMENT
cnt := 0;
DATA
FILE OPEN
REWRITE(data);
RESET(data);
PRINT_DATA
BLOCK
PROCEDURE print_data
CNT
EXTERNAL ASSIGNMENT TO TRANSMIT
WRITELN (CNT);
CNT
EXTERNAL ASSIGNMENT TO RECEIVE
READLN (cnt);
CNT < 10
T
WHILE
WHILE cnt < 10 DO
BEGIN
END;
F
CNT = 5
T
EITHER
IF cnt = 5
THEN
BEGIN
END
ELSE
BEGIN
END
END PROGRAM
END.
END BLOCK
END;
END DIGRESSION
END;

FIGURE 1: FPL Icons
with Pascal translations

FPL currently runs on computers in the IBM-PC family. The FPL programming environment is an interpreter/compiler which presents the programmer with a means for interactively entering, editing and executing graphically represented programs. The user enters an FPL program icon by icon, and can view the map-like drawing of the program at any time during entry. Upon completion, the FPL program is translated into compilable Pascal which is run via *TURBO* Pascal. Because the FPL compiler/interpreter produces code free of syntax errors, this translation is transparent to the novice user, who sees only his/her FPL drawing on paper, and observes the result of the execution of the program on screen.

Thus, FPL is not a computer-based flowchart to be used for planning program which must then be translated into traditional, textual languages for producing executable code. It is, rather, a language that provides programmers with a graphics-based, spatial environment for the entry *and* execution of programs.

The major difference between FPL and other programming languages is its graphical representation. Programs in classical programming languages must be read in a linear, prose-like fashion even when the action of the program is non-linear. Because of its unique spatial layout of connected symbols, FPL allows a reader to see her/his program as a map, in a format that more directly emphasizes its logical structure. Figure 2 presents a typical beginning program in FPL and, for comparison, in Pascal.

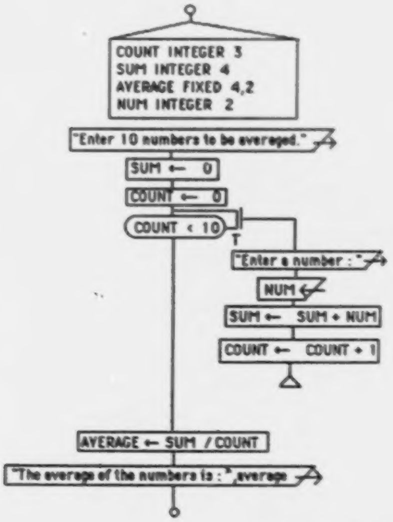
Particularly because we focus on novice learners, we feel that the graphic approach is not only desirable, but vital. Novice learners of a subject do not yet have a framework within which they can fit new learnings or concepts which appear unusual or particularly difficult. They need the clearest, most unencumbered representation possible in which to build a conceptual framework, including concrete hooks on which to hang the emerging framework.

The Research Challenge: An Agenda

The overall objective of our research agenda is to determine whether for some novice learners a graphic representation of conceptual material is more effective than is a strictly textual representation. As Shneiderman (1980, 1986) has pointed out, each small piece of research fits as a tile into the mosaic of a large research agenda. To fully understand the role and effect of the graphic alternative, we must continue to ask the questions and fit our answers, supported by empirical results, into the whole. Following is a brief description of some of the work at Teachers College, both currently underway and in planning.

The Effect of Graphic Representation

Bugs in programs written by novices. Our early studies of the effect of FPL's graphic representation focused on the presence of bugs in programs written by novices. In two different studies (Cunniff, Taylor & Black, 1986; Cunniff Taylor & Taylor, 1987) we studied programs written in FPL



```
PROGRAM test(INPUT,OUTPUT);
 VAR
   count : INTEGER;
   sum : INTEGER;
   average : REAL;
   num : INTEGER;
 BEGIN
  WRITELN('Enter 10 numbers to be averaged.');
  sum := 0;
  count := 0;
  WHILE count < 10 DO
    BEGIN
     WRITELN('Enter a number : ');
     READLN(num);
     sum := sum + num;
     count := count + 1;
    END;
  average := sum / count;
  WRITELN('The average is : ',average:4:2);
END.
```

FIGURE 2: FPL PROGRAM
and PASCAL TRANSLATION

and Pascal[1] and analyzed the types and frequency of non-syntactic (logical) bugs. We found that, although there seem to be some types of bugs that are language-independent and seem to be more closely allied to the focus of programming instruction, there are some bugs that appear to be clearly affected by the language used by the student. Specifically, bugs relating to misplaced statements and missing "update" assignments were found to be less frequent in programs written in FPL than in Pascal. These studies suggested that novices were better able to "see" where certain statements should be placed given the spatial layout of an FPL program. It is also possible that in learning elementary programming plans (Soloway, 1986), the spatial layout of FPL makes the placement of plan elements such as variable updates and running totals easier to remember.

Logical bugs such as the misuse of sentinels for the control of a loop and failure to check for valid data were not affected by programming language. These bugs seem to be the result of misconceptions about programming and do not seem to be affected by the language used by the student. We suggest that the misconceptions may be at least partially related to misunderstanding of the role and the knowledge of the programmer versus the user. Thus, the research has pointed out some important issues for consideration in the teaching of programming.

Program Comprehension. Another series of studies, currently underway, is investigating whether novices, who are learning both FPL and Pascal, comprehend programs in one language or the other more accurately and/or more rapidly. Subjects viewed a series of short program segments written in both FPL and Pascal, and answered comprehension questions. While accuracy is marginally significantly affected by language, response time for FPL program is significantly more rapid regardless of the type of program segment or the type of question asked.

These results suggest that the graphic representation helps novices assess certain aspects of a program segment more rapidly and accurately. For novices, comprehension is a critical aspect of learning programming. It is integrally involved in program construction and debugging. Helping novices with the incremental steps involved in the larger task we refer to as *programming* will certainly help allay frustration and avoid failure. More rapid and accurate comprehension will certainly be a step in the right direction, affecting both program writing and, most certainly, program debugging.

Several further studies of program comprehension, each focusing on a different aspect of the question, are in progress. One will investigate whether comprehension is more rapid when the subject is told the goal of the program before reading the actual program. Another will compare comprehension of programs that have deeply nested structures with programs that lack such nesting.

Investigation of learner characteristics. The global question regarding graphics versus text is whether or not graphical representation is a viable alternative for representing concepts and knowledge; for educators, the most critical aspect of that question is, *for whom is this alternative important?* We posit that while there may be some learners for whom graphic representation *is not* helpful, at the same time, there are those for whom it is, for whom the traditional textual focus is not helpful, and may even be counterproductive.

Our research focus, therefore, is turning toward the identification of characteristics and situations that affect a learner's interaction with, understanding of, and ability to process different representations of the same material. We are investigating whether or not learners with high visual aptitude benefit more from a visual/graphic representation than do learners with low visual aptitude, or conversely, whether learners with high verbal aptitude process textual material more aptly than do those with low verbal aptitude. Early results suggest that high visual aptitude correlates with more rapid response time on comprehension of programs written in FPL, but that verbal aptitude does not correlate with more rapid comprehension of Pascal programs.

Comparison of Graphic Versus Textual Symbolism

Children's understanding for icons. How rapidly can children learn to interpret icons? Do they remember icons better than words, and can they use them more efficiently to improve performance? Does the type of icon matter? Lodding (1983) discusses a classification scheme for icons, proposing that icons can be representational, abstract or arbitrary. Investigation of whether different icon types and specific icons are completely "context sensitive," as Lodding suggests (1983, p. 15), and whether children of various ages can use and understand icons effectively both in computing environments and in other non-computing educational situations is necessary. We are planning a series of studies to investigate these and related questions, and encourage others interested in visual communication to empirically address this important issue.

Long term memory of icons. We have been using FPL in our introductory programming courses for several years. The students in these courses had virtually no previous experience with computing. We recently conducted an informal survey of many students who had studied FPL over the past four years, to see how well they remembered the FPL icons and their meanings.

We found that for many the icons triggered accurate memory of programming functions, but not of the actual icon names. It seems that the icons, once learned, brought back the meaning of programming constructs while their corresponding textual labels were lost. This suggests that the graphic representation may help in the long term understanding of programming concepts. We plan a study of programmers who initially learned FPL and then went on to program almost exclusively in Pascal. The study will investigate whether these programmers can comprehend and debug programs in FPL or Pascal more rapidly and accurately.

Development of an environment for comparative study of language features. Comparative studies of programming languages, although often informative about specific issues, share a substantial flaw. When comparing languages, we are unfortunately not, in most cases, comparing only the relevant aspects of the languages, but also the distinct environments in which the languages are embedded, through which the user enters, edits and executes programs. The result is that the environmental differences may overwhelm the subtle language differences and muddle the results. We have recognized this problem through the studies we have already conducted, and are designing a series of studies to overcome it.

Effective analysis of the graphic effects of a language such as FPL requires that we compare students' performance with other students who are using an environment that is identical in all respects *except for the graphic representation.* We are, therefore, preparing a research version of FPL that will replace the graphic prompts and the icons with a traditional "key word" approach, while maintaining all other aspects of the environment -- for example, the on-line help and error systems and the interpreter/compiler approach to syntax error-free programming.

Concluding Comments

This paper argues that educators should use the computer to implement graphic alternatives to the traditional textual representation of concepts. It illustrates the argument, including the need for research to evaluate graphical representation, by reference to FPL, a graphically represented programming language used to teach novices at Teachers College, Columbia University. Research underway is summarized, comparing FPL and Pascal student performance in debugging and comprehension tasks.

References

Arnheim, R. (1969). *Visual Thinking.* Berkeley: University of California Press.

Brooks, F. (1975). *The mythical man-month.* Reading, MA: Addison-Wesley.

Cunniff, N., Taylor, R. P., and Black, J. B. (1986). Does programming language affect the types of conceptual bugs in novices programs? A comparison of FPL and Pascal. *Human Factors in Computer Systems: Proceedings of CHI'86,* 175-182.

Cunniff, N., Taylor, R. P., and Taylor, S. (1987). The effect of programming language on the conceptual bugs in novices' programs: A comparison of FPL and Pascal. In B. W. Hamill, R. C. Jernigan & J.C. Boudreaux, (Eds.), *The Role of Language in Problem Solving II,* NY: North-Holland Publishers (pp. 391-407).

Dear, B. L. (1986). Artificial intelligence techniques: Applications for courseware development. *Educational Technology* 26, 7-15.

Gardner, H. (1983). *Frames of Mind.* New York: Basic Books.

Grafton, R. B., Ichikawa, T., (Eds.) (1985). Visual programming [Special issue]. *IEEE Computer.*

Huggins, W. H. & Entwisle, D. R. (1974). *Iconic Communication.* Baltimore, MD: The Johns Hopkins University Press.

Johnson, W. L., Soloway, E., Cutler, B. & Draper, S. (1983). *Bug Catalogue I.* (Tech. Rep. No. 286). New Haven, CT: Yale University, Department of Computer Science.

Lodding, K. N. (1983). Iconic Interfacing. *IEEE Computer Graphics and Applications,* 18:8, 11-20.

Maruhnic, P., Lucas, T., Singer, A., Conway. M., Huras, J. (9184). *Macintosh Pascal* [Computer program]. Lexington, MA: Think Technologies.

Myers, B. A. (1986). Visual Programming, programming by example and program visualization: A taxonomy. *Human Factors in Computer Systems: Proceedings of CHI'86,* 59-66.

Morrison, P. & Morrison, E. (1961). *Charles Babbage and His Calculating Engines: Selected Writings by Charles Babbage and Others.* New York: Dover Publishing, Inc.

Polya, G. (1957) *How To Solve It.* New York : Doubleday.

Raeder, G. (1985, August). A survey of current graphical programming techniques. *IEEE Computer,* 11-25.

Ragan, T. J. (1986). Never done: Analyze learners. *Educational Technology,* 26, 25-26.

Schneyer, R. (1984). A survey of graphic algorithmic representation techniques. *Interface,* Spring, 38-48.

Shneiderman, B. (1980). *Software Psychology.* Cambridge, MA: Winthrop Publishers, Inc.

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *IEEE Computer,* 16:7, 57-69.

Shneiderman, B. (1986). Empirical studies of programmers: The territory, paths and destinations. In E. Soloway & S. Iyengar, eds., *Empirical Studies of Programmers,* Norwood, NJ: Ablex Publishers, (pp. 1-12).

Shneiderman, B., Mayer, R., McKay, D. & Heller, P. (1977). Experimental investigations of the utility of detailed flowcharts in programming. *Communications of the ACM,* 20, 373-381.

Spohrer, J. C., Soloway, E. & Pope, E. (1985). Where the bugs are. *Proceedings of CHI'85,* 47-53.

Spohrer, J. C., Pope, E., Lipman, M., Sack, W. Freiman, S., Littman, D., Johnson, L. & Soloway, E. (1984). *Bugs in novice programs and misconceptions in novice programmers.* (Tech. Rep.). New Haven, CT: Yale University, Department of Computer Science.

Soloway, E., (1986). Learning to program = Learning to construct mechanisms and explanations. *Communications of the ACM*, 19:9, 850-857.

Taylor, R. P. (1982). *Programming primer.* Reading, MA: Addison-Wesley.

Taylor, R. P.(1985). *FPL: Graphical representation of classical programming.* Unpublished manuscript, Teachers College, Columbia University: Department of Communication, Computing and Technology in Education.

Taylor, R. P., Cunniff, N. & Uchiyama, M. (1986). Learning, research and the graphical representation of programming. *Proceedings of the Fall Joint Computing Conference, 1986*, 56-63.

-- Notes --

1. See Johnson, Soloway, Cutler & Draper, 1983; Spohrer *et al.*, 1984; Spohrer, Soloway & Pope, 1985 for a description of the Pascal studies on which our research was based.

THE VALIDITY OF PROGRAMMING ABILITY TESTS

David M. Olson, Ph.D.

Western Oregon State College
Monmouth, Oregon 97361

ABSTRACT

This study evaluated the validity of measures of programming ability in an introductory computer science course. The sample included 30 students enrolled in Introduction to Computer Science, CS 211, Winter Term, 1985, at Western Oregon State College. Several of the programming abilities were assessed on a paper-and-pencil final exam. They included: (a) knowledge of language rules (GRAMMAR); (b) ability to read programs (READING); and (c) ability to write logically and grammatically correct programs (WRITING). A second set of programming abilities was assessed with a hands-on final, scored holistically and analytically.

The correlation between students' holistic and analytic scores on the hands-on final was moderate ($r = .66$), indicating that the two scoring methods measure somewhat different attributes of students' programs. Students' reading and writing scores on the paper-and-pencil final correlated more highly with the hands-on final when it was graded analytically ($r = .66$ and $r = .35$, respectively) than when it was graded holistically ($r = .45$ and $r = .35$, respectively).

The results suggest that hands-on final exams rated holistically provide the most valid measure of students' programming ability. Problems with the method, however, include high test anxiety in students and lack of specific diagnostic feedback to help students improve their programming ability. Both of these problems can be overcome.

INTRODUCTION

Typically, students' progress in computer science courses is measured using "paper-and-pencil" final exams. Items in these tests are those traditionally used to measure a construct, sometimes called programming proficiency, which is the ability to write a computer program to solve a problem. Lemos (1980) identified three aspects of programming proficiency that presumably can be measured with a paper-and-pencil test: (a) knowledge of language rules (GRAMMAR); (b) ability to read programs (READING); and (c) ability to write logically and grammatically correct programs (WRITING).

Some question exists as to the validity of these programming tests as a measure of students' proficiency in writing computer programs.

It must be remembered that the ability to perform well on a programming exam and the ability to write programs are not the same variable. The programming test is a surrogate measure of programming ability and is assumed to be a good measure of that variable but, in fact, its ability to predict programming success has yet to be determined (Cheney, 1980).

Variables

In this study, students' abilities to solve problems using a computer were measured using two types of final exams. The first was a traditional "paper-and-pencil" final exam given at the end of the term. The paper-and-pencil final was used to assess students' performance on the variables: READING, WRITING, and GRAMMAR.

The second type of exam was called the "hands-on" lab final. It measured students' ability to write and run a computer program. The test was taken in a computer lab setting under conditions approximating real-world computer programming. Students were able to use the computer and reference materials in completing the hands-on final. The hands-on final was used to assess students' performance on the variable, PROFICIENCY. The relationship between the paper-and-pencil final variables and proficiency variables was explored to improve our understanding of the extent to which the paper-and-pencil final exam measures actual programming ability and its value as a measure of programming ability.

Research Hypothesis and Questions

The following research questions were addressed in this study:

1. To what extent is student performance on measures of computer programming that involve READING, WRITING, and GRAMMAR intercorrelated?

2. To what extent is student performance on measures of READING, WRITING, and GRAMMAR correlated with their performance on measures of computer programming PROFICIENCY?

LITERATURE REVIEW

Methods of Measuring Programming Proficiency

A measure of programming proficiency should provide instructors with valid information on how well students can write programs (Haas & Hassell, 1983). Several studies have examined the different measures of programming ability commonly used in final examinations (Cheney, 1980; Haas & Hassell, 1983; Lemon, 1980; Olson, 1986; Salisbury, 1986; Van Meer & Dodrill, 1983). Lemos (1980) and Haas and Hassell (1983) observed that the WRITING portion of a programming test usually is quite different from actual programming. In actual programming, students are encouraged to use methods such as pseudo-coding to fully conceptualize the problem before attempting to code the solution. Students who have mastered the desired method may be penalized by the lack of time in a test situation. They may be forced to begin coding without using the intermediate steps.

Lemos (1980) described traditional measurement approaches to test programming ability, including approaches to measure GRAMMAR, READING and WRITING in teaching Cobol and Fortran to college students. The 1980 study evaluated the data from the student tests in two earlier studies. One study used Cobol and the other used Fortran.

Lemos found a significant correlation between reading and writing in both the Cobol study ($r = .50$) and the Fortran study ($r = .54$). The correlation between grammar and reading was higher in the Fortran study ($r = .60$) than in the Cobol study ($r = .33$), although both were statistically significant. The correlation between grammar and writing ($r = .15$, Cobol; $r = .12$, Fortran) was not significant.

Van Meer and Dodrill (1983) concluded that the ideal examination procedure for a programming course would be a hands-on final that includes writing and running a program at a terminal. Lemos discussed the disadvantages of a hands-on final given in a restricted time period:

> Students are forced to work in an artificial environment . . . . They are very likely to start coding before they have fully conceptualized the problem. The restricted amount of time that students can allot to pseudo-code, flowchart or walk-through their program can penalize students that have developed the desired trait of thoroughness (1980).

Analytical Scoring. Both the paper-and-pencil final and the hands-on final in the present study required students to write computer programs. A reliable method for scoring these programs was important to the success of the study. Miller and Peterson examined the literature seeking an analytical method for this purpose. They concluded that, "little research has been devoted to designing a better grading system for computer programs" (Miller & Peterson, 1980). Miller and Peterson did find several analytic methodologies for grading students' programs. They developed their own method of scoring students' programs based on two methods found in the literature. Students who meet minimum requirements receive 80% of the possible points. The remaining 20% is awarded to students who put extra work into the area of program style and output clarity.

Holistic Scoring. Little research concerning holistic techniques of scoring students' computer programs has been done. Related research, however, has been done in the scoring of written essays. Mullis stated that holistic scoring is:

> . . . based on the theory that a piece of writing is greater than any of its parts and that English teachers can recognize good writing when they see it. In holistic scoring, readers are asked to make a single, global quality judgment about each paper, reading rapidly for total impression. They purposely do not focus upon particular aspects of a paper such as organization, mechanics or ideas (Mullis, 1984).

METHODOLOGY

Research Design

The study employed a correlational design to investigate the relationship between students' paper-and-pencil final exam scores on measures of WRITING programs, GRAMMAR in written programs, READING programs, and programming PROFICIENCY.

Sample

The sample for this study was comprised of students enrolled in Computer Science 211 (CS 211) in Winter term of 1985 at Western Oregon State College (Western). CS 211 is the first required computer course taken by students majoring in computer science. Students majoring in business also are required to take the course. The curriculum is micro-computer based and differs in this way from programs at Oregon's three universities.

There were three sections of the class, and all students were asked to participate in the study. Thirty students completed both sets of tests (paper-and-pencil, and hands-on finals) included in the study.

Measures

Two groups of measures were administered. The first was a hands-on final used to measure students' programming PROFICIENCY. The second was the paper-and-pencil final exam used to measure students' performance in the areas of READING, WRITING, and GRAMMAR in programming.

An analytic scoring guide (Appendix 1) was prepared using the technique outlined by Miller and Peterson for analytical scoring. In addition, the hands-on final was evaluated holistically using the holistic scoring guide (Appendix 2). The holistic guide and scoring methods were based on those used for grading English essays (Mullis, 1984).

Before scoring the programs, the holistic and analytic scoring guides were reviewed by the two raters to derive a common meaning. In addition, anchor programs exemplifying each of the five levels listed on the holistic scoring guide were identified. Actual rating of each program was by two scorers, who worked independently, so inter-rater reliabilities could be calculated (Analytic, r = .73); Holistic, r = .56). The two holistic scores for each program were combined to hold random errors to a minimum. The same procedure was used with the analytic scores for each program.

RESULTS

Research Question One

Table 1 shows the intercorrelation between the variables READING, WRITING, and GRAMMAR. The only high correlation observed was between READING and WRITING. GRAMMAR correlated low with READING and WRITING.

Table 1. Correlation of Measures of Programming Proficiency on the Paper-and-Pencil Final

	READING	WRITING	GRAMMAR
READING	XX		
WRITING	.67**	XX	
GRAMMAR	.24	.28	XX

*p = .05

**p = .01

N = 30

Research Question Two

Table 2 shows the correlation between the paper-and-pencil final variables and the hands-on final variables. The READING and WRITING variables explain large amounts of variance in the analytic scores on the hands-on final but lesser amounts in the holistic final.

Table 2. Correlation of Measures of Programming Proficiency and the Hands-on Exam

	Hands-on Exam	
Written Final Exam	Holistic	Analytical
READING	.45**	66**
WRITING	.35*	.69**
GRAMMAR	.36*	.29

*p = .05

**p = .01

N = 30

DISCUSSION

Holistic and Analytic Scoring

The moderate magnitudes of the correlation (r = .67) indicate that the two scoring methods, analytic and holistic, sometimes measure different characteristics of a computer program. This phenomenon was explored by examining sample programs where the scores varied the most from one grading method to the other.

Programs with the greatest differences between holistic and analytic scores generally received a high score (B or better) when graded holistically, but a failing grade when score analytically. One such program received an analytic score of 38% and a holistic score of 80% (four out of five possible points from both graders). It received zero points in the two analytical scoring areas of program execution and planning, and low scores due to lack of prolog comments and procedures which performed multiple tasks. In none of the areas rated on the analytical scoring guide did the program receive a score higher than two.

The students' discrepancy in analytic and holistic scores could result from the time limit on the hands-on final, rather than the lack of programming ability. Examination of one hands-on final indicates that the algorithm was cor- rect but the student had not been able to successfully correct the Pascal syntax. This resulted in a program that could not compile. Consequently, no points were received in the areas dealing with style and execution. One could speculate that the lack of planning and modularity are what caused this student to be slower than other students. However, any student whose first attempt at a solution was incorrect could find themselves in the same situation.

Paper-and-Pencil Final and Hands-On Final

Students' scores on the syntax (grammar) section of the paper-and-pencil final had low correlations with their scores on the hands-on final (r = .36, Holistic; r = .29, Analytical). This result is not surprising for two reasons: (a) students were allowed to use reference materials to check proper syntax during the hands-on final but not during the paper-and-pencil final; and (b) in compiling a program the Turbo Pascal compiler points to and identifies syntax errors. The correlation between scores on the syntax section and the hands-on final might have been higher if students had not been able to use reference materials or if the compiler and operating system were less user-friendly.

Scores on the reading and writing sections of the paper-and-pencil exam had high correlations with students' scores on the hands-on final when graded analytically (r = .66 and r = .69, respectively).

In preparing the analytical scoring guide and the paper-and-pencil final, skills required in successful programming were examined. These skills, as tested on a paper-and-pencil final, were specified as the ability to read, write, and correct the syntax of computer programs. Similarly, the analytical scoring guide lists several attributes of a good program. Since both methods of measuring students' programming ability involved assessment of particular skills involved in programming, it seems reasonable that they would be highly correlated.

Holistically-Scored Hands-On Tests

In an introductory computer science course, holistic scoring could have certain disadvantages over analytical scoring. In analytical scoring, low subscores suggest areas where students need to improve. When programs are scored holistically, however, students receiving a low score may not know how to improve their programs. This weakness in holistic scoring can be overcome in three ways. First, in addition to the numerical score, the rater can place written comments on the program printout. These comments indicate weaknesses and strengths in the program and suggestions for improvement.

Second, the holistic rating guide should be made available to students. The holistic rating guide lists characteristics of programs receiving scores at each level. It differs from the analytical scoring guide in that the characteristics are not weighted for scoring purposes.

Third, programs receiving high scores could be posted as examples for other students. Stuents can use these programs as models for future work.

The holistic guide used in this study could be expanded to contain both major and minor characteristics of a computer program at each of five levels. The major characteristics would include those currently listed in the holistic guide. The minor characteristics might include specific references to programming format. Characteristics such as proper indenting of code might be applicable to all levels and could be listed separately.

Use of Hands-On Final

The hands-on test was chosen as the best way to simulate actual programming conditions. The major weakness was student anxiety resulting from the three-hour time limit imposed on the testing conditions and students' lack of experience with hands-on exams.

For most of the students in the study, this was their first experience with a hands-on computer test of any kind. Many of the students went directly to the terminals without any pre-planning even though the analytical scoring guide (which assigns points for pre-planning) was reviewed with the students prior to the test.

This anxiety could be reduced by allowing students more opportunities to develop skills for taking this type of test. These opportunities could take the form of hands-on labs, hands-on midterms, and in final exams in several computer science classes.

Students have already developed these skills in other disciplines where in-class essay exams are used in lieu of take-home exams or research papers. In order to survive the in-class essays, students must modify the techniques used in writing larger assignments.

Since the study, several courses at Western covering applications programs have gone to multiple, hands-on evaluations of student performance. Students in these courses seem to prefer the opportunity to show their ability in the lab rather than to be tested by a conventional paper-and-pencil exam.

Results of this study suggest that, if holistically scored hands-on exams are indeed the criterion for determining programming ability, the way students' programming ability is measured must be changed. Neither paper-and-pencil exams nor hand-on lab exercises, graded by using an analytical checklist, are a valid substitute.

REFERENCES

Cheney, P. (1980). Cognitive style and student programming ability: An investigation. *AEDS Journal*, *13*, 285-293.

Haas, M., & Hassell, J. (1983). A proposal for a measure of program understanding. *Communications of the ACM*, *26*, 7-13.

Lemos, R. (1980). Measuring programming language proficiency. *AEDS Journal*, *13*, 261-273.

Miller, N., & Peterson, C. (1980). A method for evaluating student written computer programs in undergraduate computer science programming language courses. *SIGCSE Bulletin*, *12*, 9-11.

Mullis, I. (1984). Scoring direct writing assessments: What are the alternatives? *Educational Measurement: Issues and Practice*, *3*, 16-18.

Olson, D. (1986). *The validity of programming ability used in the introductory computer science course*. Unpublished doctoral dissertation, University of Oregon.

Salisbury, M. (1986). *Predicting success in the introductory computer science course*. Unpublished doctoral dissertation, University of Oregon.

Van Meer, G., & Dodrill, W. (1983). A comparison of examination techniques for introductory computer programming courses. *ACM SIGCSE Bulletin*, *15*, 34-38.

APPENDIX 1

Program Evaluation Sheet

I. Planning [20%]

1. _____ Problem Specification (Problem statement which defines INPUT, OUTPUT, and PROCESSING REQUIREMENTS) (10%).

2. _____ Evidence of pre-planning (Algorithm, Pseudo code, or Structure Chart) (20%).

II. Programming Style [40%]

1. _____ Properly indented and formatted based on nesting depth (5%)

2. _____ Properly commented including prolog comments for main programs and procedures (5%)

3. _____ BEGINs and ENDs are labeled (5%)

4. _____ Use clear mnemonic names for variables (5%)

5. _____ Use clear mnemonic names procedures (5%).

6. _____ Uses only local variables and passed parameter (5%)

7. _____ Procedures perform only one specific task (10%)

8. _____ Procedures are designed so they can be reused (5%)

III. Program Execution [20%]

1. _____ Program executes properly and produces correct results (10%)

2. _____ Program provides for error checking for abnormal input and output (20%)

IV. Synthesis [20%]

1. _____ For those programs that put it all together exceptionally well or go beyond the basic assignment (20%)

APPENDIX 2

Holistic Scoring Guide
Marathon Program

Score of 5 EXCELLENT

Programmer does all or most of the following:

1. writes a program that compiles correctly and produces output.
2. writes a program that is easy to read. The programmer has made extra efforts to add to the readability of the program. Reader proceeds rapidly (is not held up by inability to decipher variable names, comments, or block structure of the program).
3. writes a program that is completely modular (uses only local variables and parameters).
4. writes a program that uses procedures which are well designed (perform a specific task and break the program into logical pieces).
5. writes program that does not contain logic errors.

* Suggested improvements to the program would be arbitrary.

Score of 4 GOOD

Programmer does all or most of the following:

1. writes a program that compiles correctly.
2. writes a program that is readable. Program contains descriptive variables and is generously commented. Reader proceeds rapidly.
3. writes a program that is generally modular (uses only a few local variables and parameters).
4. writes a program that uses procedures which are well designed (perform a specific task).
5. writes a program that contains a minor logic error.

* A small change in the design would result in a more elegant program.

Score of 3 AVERAGE

Programmer does all or most of the following:

1. writes a program that contains a minimum of compiler errors.
2. writes a program that can be followed. The program uses either well written code or comments to aide the reader.
3. writes a program that shows modularity (uses either local variables or parameters).
4. writes a program that uses procedures which are generally well designed (perform a specific task).
5. writes a program that contains some minor logic errors.

* Several changes in the design would be required to achieve an elegant program.

Score of 2 BELOW AVERAGE

Programmer does all or most of the following:

1. writes a program that contains a large number of compiler errors.
2. writes a program that is difficult to follow. Reading the program is difficult due to inability to decipher variable names, comments, or block structure of the program).
3. writes a program that lacks modularity (uses a large number of global variables).
4. writes a program that uses procedures which perform unrelated tasks.
5. writes a program that contains a major logic error.

* A major revision of one or more procedures would be required to achieve a logically correct program.

Score of 1 FAILING

Programmer does all or most of the following:

1. attempts to use reserved words or constructs that are not part of the Pascal language.
2. writes a program, part of which is impossible to follow.
3. writes a program in which signs of modularity are absent.
4. writes a program whose tasks are assigned in a random manner.
5. writes a program that contains several major logic errors.

* In order to achieve a logically correct program it would be easier to start from scratch than to use the current program.

COMPUTER ADAPTIVE TESTING

Thomas H. Saterfiel, *Chair*
Mississippi State Deptartment of Education
Jackson, MS

Sharon Johnson Lewis, *Discussant*
Detroit Public Schools
Detroit, MI

ABSTRACT

This session will discuss Computer Adaptive Testing, which is one of the topics on the cutting edge of new developments in educational measurement. At this time very few examples exist of practical applications of the concept, but much research has been done on the theory.

The panel will discuss three of the major efforts in the nation to implement CAT in a practical basis.

PANELISTS

Practical Computerized Adaptive Testing in the Classroom
James R. McBride
The Psychology Corporation
San Diego, CA

What It Will Take For CAT To Become A Reality In School
Michael H. Kean
CTB/McGraw-Hill
Monterey, CA

Using CAT To Produce The New ASVAB
Frank Vicino and Jules Borack
Naval Personnel Research and Development Center
San Diego, CA

SPONSOR: AERA

PREVIEW CENTERS AND PUBLISHERS: A COOPERATIVE VENTURE

Vicki Smith Bigham, Chair
Consultant, ICCE Past President
Houston, TX

ABSTRACT

Educators and software publishers have had, for a number of years, a dilemma regarding the circulation of software for preview. Educators want and need to look at software prior to purchasing. But software publishers cannot possibly mail a program to every educator who wants to preview it.

Regional preview centers were set up several years ago in order to solve this problem. They would provide an opportunity for educators to preview software without spending money and for software publishers to distribute reasonable numbers of products for preview without worry of copyright infringement.

What has happened, in many cases, is that even though regional centers have been established, teachers continue to request preview copies of software. Because the centers have varying procedures and guidelines as to how they are maintained, publishers have difficulty identifying those centers that are more appropriate than others for handling preview materials. These concerns are voiced whenever this topic is discussed.

During the past year, there has been a continuing forum between software publishers and representatives from statewide or other large preview centers to discuss some of the issues. What determines "the best" of centers? What criteria could be used for identification of such? How should a preview center operate?

A list of 24 suggested criteria was developed for ranking among 27 respondents, 17 from software publishing companies and 10 from preview center representatives. A subsequent survey was conducted for a broader audience on the10 most critical items for the first list. Results have been compiled and will be shared in the session.

Presented will be a drafted position paper on suggested guidelines for software publishers, software preview centers and schools. Panel members will respond to the draft and offer perspectives on the current state of the question.

PANELISTS

Marge Kosel
Sunburst Communications, Inc.
Pleasantville, NY

Ann Lathrop
San Mateo County Office of Education
Redwood City, CA

Bodie Marx
Scott, Foresman and Company
Glenview, IL

SPONSOR: ICCE

KINDERGARTENERS DISCOVER ROBOTS

Elizabeth Anne Viau

Department of Education
Coker College
Hartsville, SC 29550

Abstract

Twenty kindergarten children were intensively observed as they played with seven types of robotic toys. Problem-posing, theory-building, problem-solving, and sex-differences in play with the robots, were noted. When the children had explored the physical capabilities of the robots, they modified their cognitive schemas, and created new, experimental schemas, to facilitate the incorporation of the robotic toys into their play.

As the fruits of technology take their places in our lives, adults reach out to explore the potentialities of the new objects through study and speculation. They relate new items to the many things that they already know, and search their repertoire of paradigms for those that will suggest appropriate uses and applications for new tools.

When these tools are taken into the educational realm, however, we find that young children have a different style of learning. They have small knowledge bases and comparatively few experiences; their worlds are full of new tools and toys, and their thought processes are tied to concrete objects. These children embrace the unknowns in the world, not through formal study, but through play. Play is tentative, experimental, imaginative; it explores, probes, rearranges and asks, "What if?" In play, it is all right to be mistaken; not knowing presents no threat. Curiosity and enjoyment mark the learning that occurs through play; fantasy and "hands on" experiences are combined in activities that go beyond the boundaries of the known.

Before introducing technological innovations to the classroom, adults analyze and plan, fitting the new tools into the curriculum. Observations of the natural experimentation of young children may also be instructive; children may see new tools in ways that adults do not. Before small robots are relegated to curricular niches, it may be fruitful to determine what young children will do with robotic toys spontaneously.

The study that this paper will describe observed children at play. Robots were placed in a kindergarten and made available to the children during free play periods. The kindergarteners' spontaneous activities with the robots were observed and recorded; the notes of the observations were analyzed to discover how young children would use robots and integrate robotic toys into their experience.

Background

Robots are important because they represent the first steps of the computer into our three-dimensional world. The computer, so far, has required children to adapt to it. Young learners have had to struggle with input devices and to accept the static, two-dimensional medium of the screen. They have been constrained to stay in one spot. Robots, however, are more accessible; they can be handled, moved, lifted; and programmed to go to the child.

Chaille & Littman (1985) note that children who learn with computers are working with processes that can be repeated exactly. Software that extends the real world observations that children make, by slowing down or repeating something that is difficult to see, can make cause and effect relationships explicit. Children can then concentrate on hypothesis formulation and testing; learning becomes concerned with understanding a process, rather than coming up with a particular, correct answer (Chaille & Littman, 1985).

Kamii and DeVries (1978) identify four conditions that physical activities should satisfy if the activities are to stimulate children to think about processes as they play. These conditions are that the child must be actively able to cause change or movement in the objects; different actions by the child must produce different effects, the effects must be observable and the effects must be immediate. Robots have the potential to satisfy these conditions, and to do so in ways which can be "replayed" precisely if desired. The numeric programming requirements of some of the robots make quantification of the results easy.

At present, robotic toys are becoming available in the stores, but these toys are new to many children. Big Trak has been used with number lines and as a precursor to Logo (Keller, 1982; Keller, 1983). Robotic cars have been used to stimulate problem solving (Moore, 1985). These pioneering studies report high enthusiasm on the part of the children; however, the potential of the robot as an educational tool is still being explored. Work with the computer, however, has suggested some promising directions.

The Children and Their Classroom

The subjects of the study were twenty children: nine girls and eleven boys, ranging in age from four years, four months to six years, two months. The median age of the children was four years, eleven months. The children were enrolled in an all-day private kindergarten five days a week. They came from middle class homes.

The class was taught by two teachers, one of whom was with the class all day, while the other was present in the mornings. The classroom was a rich environment, with many construction materials and props for dramatic play. The teachers encouraged creativity and experimentation.

The Robotic Toys

Eight robotic toys were used in the study: they were the Space Tank, the Police Car, the Space Robot, the Robotic Banker, the Armatron, the Petster, and two Compurobots. The robots were chosen on the basis of suitability for the average classroom: they were battery-operated, small enough for children to pick up and carry easily, and priced from four to forty dollars.

Robots with "Visible Programs"

The Space Tank, the Police Car, and the Space Robot, all from the Radio Shack, share a common type of guidance system. Each robot comes with a set of six plastic cams, shaped like little wide-brimmed top hats, which fit on revolving, six-sided posts. Beside each post is a moveable follower, which traces the outside rim of the cam; the robot's movements are governed by the contours of the cam rim. The robots run in zig-zag paths, in circles, hexagons and other shapes. Each little robot moves with a different system: the Space Robot has caterpillar treads, the Police Car is four-wheeled and steered by the front wheels, and the Space Robot has two driven wheels and a supporting follower wheel. Because of the different types of drives, a given cam will result in different patterns if placed in different robots.

Robotic Arms

Two types of robotic arms, the Armatron and the Robotic Banker, were used in the study. These robots were also from the Radio Shack. The Armatron is not programmable: it is operated by two levers. However, it is an inexpensive model of a type of robotic arm that is used in industry. The Robotic Banker is a simple, pre-programmed, pick and place robot: it picks up a penny from a slot and deposits it in the bank through another slot, then pulls a lever to cause a cylinder to spin. The cylinder has words on it, and when it stops, the word represents the banker's "advice". It repeats these simple movements every time a penny, or a finger, is put in the slot.

The Petster

The Petster is designed to be a mechanical pet. It is a hemisphere covered with polyester fur, and has a stand-up tail and a "face" with ears, whiskers, a nose, and eyes that flash when the Petster is turned on. It is programmed to respond to sounds, and has a repertoire of responses which are elicited by different numbers of claps and other ambient noises. The Petster runs on two motorized wheels and also makes sounds.

The Compurobot

The Compurobot looks like in a sphere of white plastic; the plastic is molded to suggest arms and a head. It is programmed by means of twenty-five buttons set in a pad on top of the robot. The buttons are marked with numbers, arrows indicating directions, stars for two demonstration programs, clear, go, and a multiplication command.

The Compurobot can store and execute a long string of commands. In a conventional sense, the Compurobot is the most programable of the robotic toys; the numbers on its head make quantitative comparisons possible. Two Compurobots were put in the kindergarten classroom.

The Method

The robotic toys were made available to the children on a free choice basis during periods of free play. The children were intensively observed for twenty-five hours and twenty minutes over a nine week period. The actual play time with the robots was divided into five minute segments, and notes were made for each five minute period in which each robot was used by the children. The researcher recorded which children used each robot, what the children did with the robot, and as much of the children's conversation as possible.

When the observations had been completed, the notes were studied so that patterns could be noted. Sex differences in play with the robots, problem-posing and problem-solving, and spontaneous mathematical or pre-mathematical activities were noted. The researcher also watched for indications of the children's perceptions of the robots: they might be seen as interesting objects, as toys, as living beings, or as components in a larger context of play.

Results

The children played with the robots in a number of different ways. They engaged in pre-mathematical activities, problem-posing and problem-solving. Gender differences were observed in the way that the children played with the robots. Physical interaction, manipulation, imaginative play, and the construction of schemata were all observed.

Pre-Mathematical Activities

Computers, and, by extention, robots, are often thought of as being particularly appropriate for integration with the mathematics curriculum. The researcher was alert to spontaneous pre-mathematical and mathematical activities among the kindergartners. These activities were rare, however. The children appeared to be aware that the numerals on the Compurobots affected how far the robots would run, and that selecting larger numbers resulted in the robots travelling farther. However, numbers were utilized directly on only three observed occasions during the entire study, despite the fact that the children could read and write numerals, and demonstrated rational counting in other activities.

The children enjoyed making comparisons, however. They soon decided that the Space Tank was the strongest. They raced the robots to find out which one was the fastest. They also compared the performances of the robots under a variety of circumstances.

Problem Posing and Problem Solving

Different kinds of problems were posed and solved. One of the most common was the problem of how to get a robot to run under a child's body. Children would position themselves at the spot that they estimated that the robot would traverse. They also set up the robots so that one would crash into another, or into some other toy.

Some of the problems involved balance: the children tried to get the robots to run up a number of different kinds of ramps. These ramps increased in complexity as the study progressed, until late in the study a divided ramp of two boards separated by a space was built to accomodate the caterpillar treads of the Space Tank.

Experiments were made to discover how the robots would respond to different terrains. Again, many of these experiments involved the children's bodies directly: they would run the robots over their arms and hands. Other toys, cams, and pieces of cloth were also laid out as obstacles.

Balance was investigated by balancing different objects on top of the robots and then running the robots to see what would happen. Sometimes one robot would be placed on top of another, and the children were quite delighted to discover that balancing a robot on its undercarriage would leave the wheels free, allowing the first robot to run while balanced on top of a second robot. Once these pairs would run on level carpet, ramps and other obstacles were added to the experiment.

During all these physical experiments, the children displayed a capacity for intensive observation.

They would crawl after a robot, watching the cam go around, then remove the cam and crawl after the robot again, using a finger to move the follower. They would turn the robot upside down and run it, then lie down on the floor, holding the robot, and observe the action of the robot from below. Approximately twenty-seven per cent of the play incidents lasted for longer than fifteen minutes, which suggests that the robots were able to attract and hold the attention of the kindergartners.

A different type of problem-solving involved creativity and an expansion of the context of play. For instance, the jaws of the Armatron tended to drop objects when the wrist was rotated. One day, when the Armatron was holding a "fuzzy" (a small, furry toy) one of the boys solved this problem by putting a rubber band around the grippers to hold them shut.

Gender Differences

The robots were available to the children for 2245 individual five minute periods during the course of the study. Robots were used during 999 of those periods, or 44.5 per cent. Girls playing alone, or with other girls, used fifteen per cent of the total time available for play: play by boys used twenty three per cent of the time. Only six per cent of the periods were utilized by boys and girls: boys and girls played with the robots together for only 3.25 per cent of the time. After the data had been adjusted for the different days of attendance of boys and girls, the robots were actually used by the boys for 55.5 per cent of the utilized time, and by the girls for 44.5 per cent of the utilized time.

The children also showed sex-related preferences for individual robots. The boys' favorite robot was the Space Tank, which they used more than four times as often as the girls did. The boys also used the Space Robot three times as often as the girls. The most popular robot, however, was the Petster, which the girls used two and a half times as often as the boys. The boys used the Armatron half again as often as the girls, and the Robotic Banker, the Police Car and the Compurobots were roughly gender-neutral.

Reasons for these preferences will be discussed later in this paper: suffice it to note that the furry Petster (in the role of a pet or baby) fits in well with nurturing paradigms, while the Space Tank and the Space Robot fit into paradigms familiar to boys. The robots which were gender-neutral are those which do not fit into specific cultural schema, except for the Police Car, whose very specific role may have been an obstacle to generalized play formats.

Physical Interaction and Manipulation

The children took an immediate interest in the robots, and interest in the toys remained high throughout the study. The children's interactions with the robots were very actively physical; they crawled after the robots, jumped over them, and made bridges of their bodies so that the robots could run under them. Sometimes they crawled above the robots, so that the robots ran with the children above them.

The children also enjoyed fine motor activities with the robots. During the first part of the study, the kindergartners spent time putting the cams into the Space Robot, the Police Car, and the Space Tank. Later in the study, the children played with the grippers of the Robotic Banker and the Armatron.

Imaginative Play

Play with the robots went beyond physical activity. Imaginative play was an important part of the interaction. Pretence might be as simple as making car noises as the Police Car ran, or it might develop into a complex script. Imaginative play with the robots was divided into four quarters: the base line percentage of incidents set in the first quarter had doubled by the third quarter, and remained at this level through the fourth.

The Construction of Schemata

The most striking feature of the children's play was the effort that they made to assimilate the robots into the cognitive structures of their play. After a period of exploration, during which the children discovered what the robots could do and how to make them run, the kindergartners began to play out familiar schemata, replacing some other toy with the robot. Examples of this would be "flying" the robots, i.e., carrying the robots and making "flying" motions and sounds, or "Good Guy, Bad Guy" where crashes and chases were used in familiar pursuit themes.

In the second stage, new, experimental schemata were created for the robots. Some of these schemata were tried only once or twice, and proved to have limited potential for play. Nonetheless, these were original inventions: such concepts as the "underwear switch" on a robot, or "Energy Exchange", where one robot gave energy to another, were thrusts in a genuine search for meaning.

A third stage in the construction of schemata was the invention and elaboration of complex microworlds. A number of scripts were played out in these worlds. The microworlds revolved around specific robots, and included the simple world of the Robotic Banker, the Block World which was mainly created by the boys, and the World of the Petster, which was mainly created by the girls.

The Robotic Banker

The Robotic Banker is a pick and place robot that is started by placing a penny in a shallow slot. The robot picks up the penny, moves it through a ninety degree arc, and releases it into a slot so that the coin falls through the slot and into the coin storage vault. The robot then moves the gripper over to a lever, which the gripper grasps and pulls. Pulling the lever causes a cylinder to spin. The cylinder is imprinted with several words, and it stops to reveal a single word or phrase, such as "Definitely" or "Ask again".

The children enjoyed the predictability of this unvarying routine. They soon began to invent ways to vary it, and created the script of "Fool the Banker", where they would put a penny in the shallow slot, watch the robot start its task, and then snatch the penny away, laughing gleefully as the robot continued its work. Sometimes they would use two pennies, or their own fingers, to start the robot: sometimes they allowed the robot to carry their fingers through the entire routine. During this type of play, the children engaged in animated discussion, often saying, "I fooled the Banker!" "He thinks he has a penny!" "He thinks my finger is a penny!" They also learned to read the words on the cylinder.

The Robotic Banker was also used for other experiments. Sometimes the children removed the stopper on the bottom of the coin storage receptacle, started the robot, and lay down, holding the robot above them, to observe the routine from a different point of view. They also made the bank gripper engage in pinching play with the Armatron.

The Block World

The block world was a world of patterns. In it, robotic toys were integrated with the construction materials already in the classroom. Initially, the Space Robot, the Space Tank, the Police Car, and the Compurobots were used to push each other and individual blocks; then the robots were enclosed in block structures from which they were made to escape. The children experimented to see if the robots could climb up ramps. As the study progressed, the children built increasingly complex environments for the robots: roadways of blocks were built, providing a basis for measurement. Sidewalls were placed along the roadway, providing opportunites for one-to-one matching. Ramps connected different levels of the roadway: and towers, obstacles and sudden drops were incorporated in the structure.

As the study progressed, more and more robots were included in the block world. Complex trains were constructed, trains which eventually included all the available robots, even those without wheels. The trains were generally set up as opposing trains, pushing against each other.

The World of the Petster

The world of the Petster was a world of relationships, and was created mainly by the girls. The Petster served as a focus for the play, and was treated like a pet, or a younger sibling. The Petster was talked to, kissed, stroked, fed plastic flowers, read to, and even spanked! Although the Petster is designed to respond to specific sound sequences, the ambient noise in the classroom also triggered its actions, and it appeared to be almost alive in its independence of action.

The children tied yarn and bandages on the robot, ran it with sunglasses on, and built block structures for the Petster to knock down. The Petster was also the focus of a few violent actions by some of the boys, who attacked it several times, and once even pretended to kill it. The girls were protective of the Petster.

Conclusions

The children's uses of the robotic toys in the sample studied were varied and creative. Although the most common activities were exploration and physical play, the intensity of the children's curiosity was striking, and the problem-solving and problem-posing that they demonstrated showed an experimental approach to these unfamiliar toys.

The frequency of the children's play with the robots appeared to be influenced by how easily the robots could be fitted into familiar schemata. The Block World and the World of the Petster started from a nucleus of familiar concepts; however, new and experimental scripts were also created in both these worlds.

The children in this study worked with robotic toys to discover and create meaning on many levels. Although they were given no models of "appropriate" ways to play with robots, they developed experimental scripts for all of them. Young children construct coherent cognitive models of reality from their own experiences. Robots may well join the beloved toys which stimulate the playful exercise of this power.

BIBLIOGRAPHY

Chaille, C. & Littman, B. (June, 1985) Computers in early education: the child as theory builder. In E. L. Klein (Ed) *Children and Computers*. New Directions for Child Development, pp. 19-35.

Heron, R.E. & B. Sutton-Smith. (1971) *Child's Play*. New York; Wiley.

Kamii, C. & DeVries, R. (1978) *Physical Knowledge in Preschool Education*. Englewood Cliffs, N.J.:Prentice Hall.

Keller, J.K. (1983) *Commack Public Schools Big Trak Assessment 1982-1983*. Commack: Commack Public Schools.

Keller, J.K. (1982) *Commack Public Schools Big Trak Assessment 1981-1982*. Commack: Commack Public Schools.

Moore, M. (1985) Solving problems with robots. *Conference on Computers and the Human Mind*. Eugene: CATE.

Pritchard, S.F. (1983-1984) Robots, bulldozers, and other map activities for the primary grades. *Journal of Geography*. 8.

THE RELATIONSHIP BETWEEN HOME MICROCOMPUTER OWNERSHIP AND DEVELOPMENT OF KINDERGARTEN STUDENTS

Thomas E. Franklin and Ronald Zoromski

University of Wisconsin-Stout

Menomonie, WI 54751

Abstract

The purposes of this research were to survey kindergarten students' parents to determine parents' attitudes toward the educational importance of computers to their children, the frequency of home computer ownership, the types of computer hardware and software owned, and to compare children's physical, emotional-social, mental and reading readiness development for those with and without access to home computers, as well as for some other distinguishing characteristics of children's homes. Conclusions were based on a sample of 558 students' parents (77% rate of response) and on two randomly selected samples of kindergarten students (n=80) for whom their teachers completed individual developmental progress reports. Parents believed that owning a home computer would help their children improve in school (84%) and that children should be introduced to computers prior to age five (50%). One in every seven homes had a computer, with significantly more male kindergarteners having home access than females. Analysis of variance comparisons of samples of owners' and non-owners' developmental progress reports, supplied by their teachers, showed no significant differences on any single developmental variable (34) or subscale variable (4). However, investigations of the relationships between developmental variables and various patterns of home computer use by the 40 randomly selected children from homes with computers resulted in a variety of significantly higher developmental ratings for children whose parents helped them with their computers, and children who had access to software programs in reading, math, and science.

The belief that familiarity or skill with computer technology is an essential requisite for successful schooling or career has permeated American culture. One of the indications of this belief is the fact that 93% of our secondary schools and 85% of our elementary schools have and use computers. There have been estimated to be over one million computers in our public schools. Over one-half of our secondary schools have fifteen or more microcomputers and elementary schools have an average of five or more microcomputers(Electronic Learning , 1986).

In addition to the prevalence of computers in our schools, a further testament to the perceived importance of computers to our lives is the frequency of home computer ownership. A recent estimate suggested that 16% of American homes, or one in every five, have a personal computer (Dutton et.al., 1985) that is used by an average of two to three people (Case and Daley, 1983). Home purchases represent over forty percent of the entire personal computer market (Dutton et.al., 1985) and households are becoming the main purchasers of educational software (Futurist , 1983). Research on how home computers are used has reported three predominant categories of applications: entertainment/games, education/learning aids, and efficiency applications such as word processing and spread sheet analyses (Case and Daley, 1983; Holden, 1984; and Wollsman, 1984).

Some advocates of home computer ownership have suggested that the machine can be a source of group activity, a family focal point that will make learning fun and enhance the family's educational and efficiency potential (Perry, 1982; Levine, 1983; Futurist , 1983; Wollman, 1984; and Dutton et.al., 1985). However, some of the positive claims, and indeed the general public's valuing of computers, may not be validated through research or actual experience, but may be partially based on marketing techniques used to sell hardware and software. Critics and some researchers have described a variety of concerns about the values of computer interaction which are remarkably similar for the school and home contexts. For example, software has been criticized as being developmentally or educationally inappropriate (Bork, 1984), used mostly for drill and practice as opposed to facilitating creativity or judgement (Becker, 1984). Further criticisms have been that computers are difficult to use effectively, parents or teachers are usually not well prepared or trained to teach computing effectively, and computers may aggravate some existing problems facing education, such as gender equity (Brady and Hill, 1984). It has been reported that 93% of home users are males (Holden, 1984).

Other issues which are equally valid in school or home contexts and are deserving of research attention have to do with the relationships between computer interactions and children's development. Topics which have been debated include the optimal age to introduce children to computers, whether children's motor skills are developed or hindered by computer interaction, what cognitive skills are facilitated through computer interaction, and what are the impacts of computer interaction on children's social and emotional development. Some psychologists believe that preschool age is the optimal time to introduce children to computers (Beeson and Williams, 1983; Sheingold, 1984). Others, however, feel that a child should at least be at a concrete operations stage of development (6 to 8 years old) before computer interaction will be meaningful (Brady and Hill, 1984). While it has been argued that joystick manipulation is desirable for development of eye hand coordination (Burg, 1984), there are others who believe children would be better off manipulating paint, clay, crayons or playing instruments (Sheingold, 1984; Cuffaro, 1984). While providing as rich and intellectually stimulating an environment as is possible is important to children's cognitive development, particularly prior to age 6, it appears that there is very little actual research about the computer's value in facilitating cognitive development (Hoot, 1983). There are, however, a multitude of people talking about the computer's "potential" for expanding communication and intellectual abilities of young children (Hoot, 1983; Brady and Hill, 1984; Grisworld, 1984; Williams and Williams, 1984). Furthermore, the fear has been expressed that extensive computer interaction will isolate kids, reduce their social interactions and hamper their social development (Feldman and Sears, 1970; Levine, 1983). In contrast, other research has shown children winning approval from classmates by their computer competence, demonstrating cooperative activity in problem solving or showing greater willingness to share when working with computers (Muller and Perlmutter, 1984; Swigger and Swigger, 1984; Brady an Hill, 1984; Dutton et.al., 1985).

Which perspectives are more accurate in these debates about the computer's "potential" in these developmental areas are currently unknown. "Unfortunately, nearly nothing is known about the effects of microcomputers on the development of young children (Hoot, 1983)." What does seem clear is that computers exist in vast numbers in our schools and homes, but also that the commercial software companies could be more responsive to children's educational needs and make software more tailored to children's developmental levels, and that more research is needed in these important developmental areas. The research reported below has focused on home computer use of kindergarten students and has addressed some of these issues in an attempt to document: 1) parents' attitudes about the values of home computer ownership and the importance of computer interactions for their children; 2) the frequency of ownership, types and uses of equipment and software computer owners with children have; and, 3) the relationships between home computer ownership and kindergarten students' physical, emotional-social, mental, and reading readiness development.

Method

Subjects

In this study, data were collected from kindergarten students' parents and their teachers. The first group of subjects was the parents of kindergarten students enrolled in the Eau Claire School District, Eau Claire, Wisconsin, in the Fall of 1985. Permission to survey kindergarten students' parents was obtained from sixteen of the seventeen schools in the district. The parents were surveyed to determine which kindergarteners had or did not have access to home computers, the types and uses of computers the owners had, and to assess parents' attitudes toward computers. The survey instruments were carried home by the students and returned by the students to their teachers. Of the 723 students' parents who should have received questionnaires, responses were obtained from 558 students' parents, for a response rate of 77 percent.

Based on information gathered from

the responding group of students' parents, two groups of 40 kindergarten students each were randomly selected. One group was children who had access to a home computer and the other group was children who did not have access to a home computer. The teachers of those 80 students were asked to complete developmental progress reports on them, without any knowledge of which students did or did not have a computer at home. Progress reports were provided by nineteen separate teachers, representing 32 separate kindergarten classes. The reports were evaluations of each student's physical, social-emotional, mental, and reading readiness development, and enabled comparisons of developmental variables between kindergarteners with and without home computer access, as well as some other distinguishing characteristics.

Instruments

Two separate paper and pencil questionnaires were used in this research. A take home questionnaire for the parents of kindergarten students was used to find out which students had or did not have access to home computers. This survey instrument consisted of sixteen items. A set of six questions were to have been answered by all parents who had children enrolled in kindergarten classes. The questions asked parents about their educational backgrounds, family income, attitudes about computers and their educational value, the age at which children should first be introduced to computers, and which homes had computer systems. A set of nine items were directed at only families who had home computers. The questions asked parents what types of computer hardware and software were available to their child, who in their homes used the computer, for how many hours each week their computer was used for entertainment, word processing, budgeting or as a learning aid for children, and whether a parent spent time helping their child on the home computer. This total instrument was pilot tested with five parents of kindergarten students and should have only taken respondents about five minutes to complete.

The second instrument used in this research was a developmental progress report completed by kindergarten teachers for forty children randomly selected from those who had access to a home computer and forty children randomly selected from those without a home computer. This progress report of 34 items was based on an instrument developed for use by the Independent School District 256, Red Wing, Minnesota (1979). Teachers responded to each of the items using a four point Likert-type scale of "Outstanding," "Satisfactory," "Needs Some Improvement," or "Needs Much Improvement." Six of the itmes were on physical development and measured children's motor readiness in areas such as knows left from right, dresses without assistance, uses scissors correctly, and copies simple shapes. Ten items on social development measured such areas as emotional readiness, self-confidence, participation in group activities, and consideration for others. Eighteen items involved cognitive development. These included six items on mental development, such as work readiness, listening skills, speaks clearly and plainly, works independently, and classifies basic shapes; four items on math readiness, such as understands math terms, identifies sets and number sets, and recognizes numbers and their sequence; and eight items on reading readiness, such as prints first name, recognizes upper and lower case letters, prints in sequence, and knows consonant sounds. The approximate time for the teachers to complete the progress report for each student was three to five minutes.

Procedures

Permission to conduct this research in the Eau Claire Area Schools was obtained from the Superintendent of Schools in Eau Claire, Wisconsin, and individually from each of the participating teachers. Based on the class lists from each of the participating nineteen teachers and their 32 classes, cover letters, questionnaires for parents, and return envelopes were sealed in separate envelopes with each child's name on them. Teachers were instructed to hand out the surveys to their students at the end of class, Monday, October 21, 1985, with instructions to the children to carry them home to their parents and to return them to their teachers when their parents asked them to do so (by the due-date of one week later when a researcher would return to the schools for the completed instruments). Of the 723 questionnaires carried home, 558 were completed and returned (77%). There were no followup activities to improve rate of response.

Based on students' parents' responses to the first instrument, two groups of children were identified.

One group was children who had access to a home computer and the other group was children who did not have access to a computer at home. Forty students were randomly selected from each of those two groups and developmental progress reports were delivered to the teachers of those 80 students. Each of the nineteen teachers were involved in this phase of the study. The teachers were not informed about which students did or did not have computers at home. After completing their progress reports, the teachers were instructed to seal them and leave them in the school's office for retrieval by a researcher. Comparisons between students with and without home computer access on the developmental progress report variables were a major focus of this study.

Results

This report of results is divided according to the two instruments used in this research and their associated purposes.

Parents' Survey

As has already been mentioned, 558 parents of the 723 sent questionnaires responded, for a response rate of 77%. The respondents were parents of 267 male (47.85%) and 291 female (52.15%) kindergarten students. The vast majority of respondents were female parents (N=470; 85.30%). When asked to indicate the educational level of the parent with the most years of formal schooling, it was found that a majority of parents (54%) had some college or technical college experience. Twenty-three percent of the sample had high school degrees, 1.5% had grade school only, 7% had Bachelor's degrees, 11% had Master's degrees and 3.5% had doctorate degrees. Their combined household incomes were reported in income categories. The most frequent category was $20,000-$29,999 (28%), with 21% in the $10,000-$19,999 range and 22% in the $31,000-$39,999 range. These demographic characteristics appeared to be representative of people in small and middle-sized west central Wisconsin communities.

Several questions were asked of all respondents to ascertain their attitudes about computers and their educational value. When asked, "When you first heard of computers, were you immediately interested in owning one?", 357 (64.56%) responded "NO." However, when asked if they thought that owning a computer would help their child improve in school, a resounding 463 (83.88%) responded "YES." When the total sample was asked how old a child should be when introduced to a computer, over 50% felt that children should be introduced before the age of five when they begin kindergarten. About 10% felt that two or younger was appropriate, 42% felt that three to four was best, 32% felt five to six years was best, and 17% thought children should be seven or older.

Seventy-nine of the 558 respondents reported owning a computer that was kept in their homes. This was 14%, or one in every seven, of the homes. Of those 79 homes with computers, 49 (62.03%) had male kindergarteners and 30 (37.97%) had female kindergarteners. A Chi Square Analysis of this ratio indicated that there were significantly more male kindergarten students with access to a home computer than female kindergarten students (X=7.41; df=1; p=.01).

In comparisons between computer owners and the non-owners in the sample on some of the computer attitude items, it was discovered that significantly more owners (37 of 79) than non-owners (159 of 479) were initially interested in owning a computer when they first heard of them (X=5.23; df=1; p=.05). Significantly more owners (72 of 79) than non-owners (391 of 479) felt that home ownership would help their children improve in school (X=4.77; df=1; p=.05). Also, owners felt that children should be introduced to computers at significantly younger ages than non-owners (X=25.60; df=3; p=.001). In fact, 66% of owners felt that age 3-4 was when children should be introduced, and 78% of owners thought children should be introduced before age five and kindergarten. Furthermore, there were significantly more college degrees among owners (22 of 79) than non-owners (95 of 479) (X=10.91; df=3; p=.05). However, there was no significant relationship between income categories and ownership.

When asked what brands of computers they owned, a variety were reported, with Commodore reported most frequently at 23%, followed by Atari (12%), Radio Shack (10%), Apple (6.5%), IBM (4%), and other assorted brands (43%). Four of the 79 parents reported having two computers in their homes. Joysticks were reported as peripheral equipment by 65% of the parents. Other peripheral equipment included extra disk drives (34%), printers (27%), modems (8%), and mouses (4%). The parents reported that their home computers were kept in a variety of locations, including living rooms

(29%), bedrooms (22%), dens or studies (17%), family rooms (13%), and even kitchens (2.5%).

When asked who in the home uses the computer, children under age 6 were reported most frequently (76%), followed by older children (73%), the parent completing the survey (72%), and spouse (51%). In addition, 60 of the 79 parents (81%) indicated that they spent time helping their child on the home computer. When asked how many hours per week their computers were used for various purposes, entertainment and as learning aids were reported frequently and for the most hours on average. (See Table 1 for more detail on these results.) There were no significant differences between male and female students' or among parents' educational attainment categories in regard to uses of home computers when measured according to hours per week or frequencies of homes for the purposes listed in Table 1.

In regard to the types of programs in homes for use by children, it was determined that games were the most frequently reported (83%), followed by aid in math software (60%), aid in reading software (49%), none (9%), aid in science software (8%), and other software programs (32%). Furthermore, when parents were asked the main changes that the home computer brought about for their families in a structured, multiple choice item, most parents reported "No noticeable change (70%), followed by the responses "The family spends more time together" (20%), or "Other" (10%). None of the 79 parents indicated the response "The family seems to be separated more."

Developmental Progress Reports

Each of the kindergarten teachers in the study completed developmental progress reports for students in their classes included in random samples of 40 students with computers at home and 40 students without computers at home. Not knowing which group each student fit, teachers used a Likert-type scale of "Outstanding" (1), "Satisfactory" (2), "Needs Some Improvement" (3), "Needs Much Improvement" (4), for 34 separate items, with the following subscales: Physical Development (6 items), Emotional-Social Development (10 items), Mental Development, including Math Understanding (10 items), and Reading Readiness (8 items).

Analysis of Variance statistical comparisons of the samples of owners' and non-owners' progress reports showed no significant differences on any single developmental progress report variable or subscale variable. In fact, there was a slight edge on the mean developmental ratings held by the non-owners. This may have resulted from the disproportionate ratio of female (60%) to male (40%) students in the randomly selected non-owner group compared to the owner group (30% female to 70% male).

Investigations of the relationships between developmental variables and various patterns of home computer use by the 40 children from homes with computers resulted in some significant findings. For example, comparisons of developmental progress reports for children whose parents indicated they helped them on their computers (N=31) to those whose parents did not indicate that they helped (N=5) resulted in significant differences in ANOVA's for two of the four subscale scores and the total progress report scores. (See Table 2 for these findings.)

Comparisons of kindergarten students who had entertainment software (N=34) to those without entertainment software (N-5) revealed no significant differences on any progress report variable. However, ANOVA comparisons of those who had software for Aid in Reading (N=19) to those without those types of programs (N-20) revealed eight significant differences on developmental variables. (Table 3 reports those significant findings.)

Comparisons of those with Aid in Math software (N=20) to those without those types of programs (N=19) revealed fifteen significant differences on developmental variables when examined by Analyses of Variance, including two of four subscales and the total progress report score. (See Table 4.)

Comparisons of those who had software for Aid in Science (N=4) to those without those types of programs (N=35) revealed eight significant differences on developmental variables in ANOVA's. (See Table 5.)

Discussion

The purposes of this research were to survey kindergarten students' parents to determine parents' attitudes toward the educational importance of computers to their children, the frequency of home computer ownership, the types of computer hardware and software owned, and to compare children's physical, emotional-social, mental and reading readiness development for those with and without access to home computers, as well as on some other distinguishing characteristics of children's homes. The findings of this research have been based on a relatively large sample of 558 students' parents, representing 77% of those requested to participate, and on two randomly selected samples of kindergarten students (N=80) for whom their teachers completed individual developmental progress reports.

Even though very few parents expressed an initial interest in owning a home computer when they first heard of them (35%), an overwhelming 84% of parents believed that owning a home computer would help their children improve in school. Furthermore, over 50% of the parents believed that children should be introduced to computers before the age of five when they begin kindergarten. Parents, it appeared, were "sold" on the importance of computers to their children's futures.

Seventy-nine of the respondents reported owning a computer that was kept in their homes. This was 14%, or one in every seven, of the homes. Of those homes, significantly more male kindergarten students had access to home computers than female kindergarten students (p=.01). The finding on frequency of ownership is comparable to national results (Case and Daley, 1983), and the gender inequity result is consistent with other research (Brady and Hill, 1984; and Holden, 1984).

In comparisons of computer owners and non-owners, it was discovered that significantly more owners were initially more interested in owning computers when they first heard of them (p=.05), felt that home ownership would help their children improve in school (p=.05), and felt that children should be introduced to computers at younger ages (p=.001) in comparisons to non-owners. These results suggested strong committments among owners in support of their decisions to provide their children with computers. Furthermore, there were significantly more college degrees among owners than non-owners (p=.05), but there was no significant relationship between ownership and income categories.

In regard to patterns of use, it was found that children under age 6 (76%) and older children (73%) were most frequently reported as using home computers. And, it appeared that in the majority of cases the children were not expected to manipulate their computers all on their own, since in 81% of the homes parents reported that they helped their children on the computers. These conclusions which suggested that home computer use was focused on the children were further confirmed by the findings that as entertainment and as learning aids for children were reported more frequently and for more mean hours per week than any other computer applications. Of the software programs available for children in the homes, games were reported most frequently (83%), followed by aid in math software (60%), aid in reading software (49%), and aid in science software (8%). The availability and perceived values of science software for children are issues which deserve attention within schools and by commercial software manufacturers.

In regard to the social impacts of computers in homes, most parents indicated that the presence of a computer resulted in no noticeable change in their homes (70%), 20% responded that the computer resulted in the family spending more time together, and surprisingly, none of the parents believed that because of the computer their family seemed to be separated more (10% chose an "Other" response), thereby somewhat dispelling the belief that the computer isolates people.

One of the primary purposes of this research was to determine if home computer ownership was related to development of kindergarten students. In addressing this purpose, kindergarten teachers completed developmental progress reports for random samples of owners and non-owners. Analysis of Variance statistical comparisons of the samples of owners' and non-owners' progress reports showed no significant differences on any single developmental variable (34) or subscale variable (4). However, investigations of the relationships between developmental variables and various patterns of home computer use by the 40 randomly selected children from homes with computers resulted in some significant findings. For example, comparisons of developmental progress reports for children whose parents indicated they helped them on their computers to those whose parents did not indicate that they helped resulted in significant differences for two of the four subscale scores and the total progress report scores. When parents helped their children, those children were rated significantly higher in the developmental areas of Physical development (p=.01), Reading Readiness (p=.01), and on the Total developmental score summarizing all 34 instrument items (p=.05). In other words, owning a computer is not what is important, but using it instructively is important.

Comparisons of kindergarten students who had entertainment, aid in reading, math and science software to those kindergarten students without those types of software demonstrated some interesting relationships further testifying to the importance of how computers are used in the home. There were no significant relationships between ownership of software games and any developmental progress report variables. However, ownership of Aid in Reading software was associated with significantly higher developmental ratings in eight areas in the subscale categories of Physical Development (3), Mental Development (1), and Reading Readiness (4). Ownership of Aid in Math software was associated with significantly higher developmental ratings in fifteen areas in the subscale categories of Physical Development (3), Emotional-Social Development (8), Mental Development (3), and for the Total Progress Report score. And, ownership of Aid in Science software was associated with significantly higher developmental ratings in eight areas in the subscale categories of Emotional-Social Development (6), and Reading Readiness (2). Based on the developmental accomplishments valued by kindergarten teachers, incorporated in the instrument in this study, it was demonstrated that students' interactions with different types of educational software for use by children were associated with significant differences in teachers' ratings of their students in a variety of developmental areas, often consistent with the intended purpose of the software.

This study has demonstrated the perceived educational importance of computers to kindergarten students' parents. Furthermore, it has shown somewhat conclusively that just having a computer at home is not what it takes to achieve the maximal positive impact of computer interactions. What does seem important is how those computers are used. We cannot assume that parents are automatically good teachers or well informed in how to use computers themselves. Serious attention should be given to educating those who guide our children, parents and teachers alike, in how to use computers to their best developmental advantages.

References

Becker, Henry J. (1985). Computers in schools today: Some basic considerations. *American Journal of Education, 93*(1), 22-39.

Beeson, B. S., & Williams, R. A. (1983). *The effects of gender and age on preschool children's choice of the computer as a child-selected activity.* Muncie, IN: Ball State University. (ERIC Document Reproduction Services No. ED 233 810)

Bork, A. (1984, December). Computers in education today and some possible futures. *Phi Delta Kappan*, 239-243.

Brady, Elizabeth H. & Hill, Shirley (1984, March). (California State University, Northridge) Young children and microcomputers: Research issues and directions. *Young Children, 39*,(3), 49-61.

Burg, Karen (1984, March). (University Connecticut, Child Development Labs) The microcomputer in the kindergarten: A magical, useful, expensive toy. *Young Children, 39*(3), 28-33.

Case, D., & Daley, H. (1983, May). *Personal computers: The new academic medium.* Paper presented at the 33rd Annual Conference of the International Communication Association, Dallas, Texas. (ERIC Document Reproduction Services No. ED 235 778)

Clement, Frank J. (1981, April). Affective considerations in computer-based education, *Educational Technology,* 28-32.

Computerized home education movement to take off. (1983, August). *Futurist, 17*, 69.

Cuffaro, Harriet K. (1984, Summer), (Bank Street College of Education) Microcomputers in education: Why is earlier better?, *Teachers College Record, 85* (4), 559-563.

Dutton, W., Kovaric, P., & Steinfield, C. (1985, January-March). Computing in the home: A research paradigm. *Computers and the Social Sciences, 1*(1), 5-18.

Feldman, D., and Sears, P. (1970). Effects of computer assisted instruction on children's behavior. *Educational Technology, 10*(3), 11-14.

Grisworld, Philip A. (1984, Winter). Elementary students' attitudes during 2 years of computer-assisted instruction. American Educational Research Journal, 21(4), 737-754.

Holden, C. (1984, July 20) Will home computers transform School?, Science, 225, 296.

Holt, J. C. (1983). How children learn. New York: Delacorte Press.

Hoot, J. L. (1983). Computers and very young children: Educational promise and problems. (ERIC Document Reproduction Service No. ED 235 897)

Kulik, J. A., Bangert, R. L., & Williams, G. W. (1983). Effects of computer-based teaching on secondary school students. Journal of Educational Psychology, 75, 19-26.

Lepper, Mark R. (1985, January). Microcomputers in education: Motivational and social issues. American Psychologist, 40(1), 1-18.

Levine, J. A. (1983, July). A computer in the family. Parents, 58, 53-54.

Muller, A. A.; Perlmutter, M. (1984, January). Preschool children's problem-solving interactions at computers and jigsaw puzzles. 1-30. (ED 248-013)

News: Survey maps computer use. (1986, January). Electronic Learning,12-13.

Pepe, John (1984).· Microcomputers in schools, 1983-1984: A comprehensive survey and analysis. Market Data Retrieval, Inc., Westport, CT (ISBN 0-89770-284-0)

Perry, R. L. (1982, July). School's out for summer (Home computer activities). Mechanix Illustrated, 78, 36.

Sheingold, Karen (1984, May). The microcomputer as a medium for young children. (Tech. Rep. No. 26). New York, NY: Bank Street College of Education. (ERIC Document Reproduction Service No. ED 249 923)

Smith, Penny (1984, April). Pre-schoolers learn at home, Creative Computing, 11, 81-84.

Sweetnam, G. (1982, November). Computer kids: The 21st century elite. Science Digest, 90, 84-88.

Swigger, Kathleen M. and Swigger, Boyd K. (1984, spring). Social patterns and computer use among preschool children. AEDS Journal, 17(3), 35-41.

Turkle, Sherry (1984). The second self-computers and the human spirit. Simon and Schuster.

Wellborn, Stanley N. (1980, December). What the next 20 years hold for you. U. S. News and World Report, 139.

Williams, Frederick and Williams, Victoria (1984). Microcomputers in elementary education: Perspectives on implementation. Wadsworth Publishing Co., Belmont, CA

Table 1
Mean Hours Per Week and Frequency Home Computers Were Used for Various Purposes

	$\bar{X}$	SD	N	%
Entertainment/Games	4.46	4.08	57	72.15
Learning Aid for Children	3.54	4.26	48	60.76
Home Budgeting	2.31	2.09	13	16.45
Word Processing	1.71	1.11	7	8.86
Other	5.54	4.26	13	16.45

Table 2
Means and ANOVA's Comparing Developmental Subscales for Students Whose Parents Did or Did Not Help Them on Computers

Subscale	Helped		Not Helped		F	df	p
	$\bar{X}$	SD	$\bar{X}$	SD			
Physical Development	2.06	.42	2.67	.66	7.81	1,34	.01
Emotional-Social Development	2.04	.42	2.30	.52	1.51	1,34	N.S.
Mental Development	2.03	.63	2.44	.72	1.74	1,34	N.S.
Reading Readiness	2.09	.61	2.93	.58	8.04	1,34	.01
Total Progress Report	2.06	.46	2.55	.60	4.63	1,34	.05

Key: lower score = higher rating

Table 3
Means and Significant ANOVA's Comparing Developmental Variables for Students Who Had or Did Not Have Software Programs for Aid in Reading

Developmental Variable	Reading Aid		No Reading Aid		F	df	p
	$\bar{X}$	SD	$\bar{X}$	SD			
Physical Development Subscale	1.97	.58	2.31	.41	4.36	1,37	.05
Ties Shoes	2.11	.81	2.70	.80	5.32	1.37	.05
Holds Crayons and Pencils Correctly	1.84	.60	2.25	.64	4.20	1,37	.05
Mental Development							
Work Readiness	1.79	.71	2.25	.72	4.04	1,37	.05
Reading Readiness							
Sees Likeness and Differences	1.74	.65	2.25	.64	6.15	1,37	.05
Reads Left to Right	1.89	.74	2.42	.77	4.64	1,37	.05
Prints in Sequence	2.10	.88	2.76	.90	4.94	1,37	.05
Knows Consonant Sounds	2.53	.90	3.12	.70	4.74	1,37	.05

Key: lower score = higher rating

Table 4

Means and Significant ANOVA's Comparing Developmental Variables for Students Who Had or Did Not Have Software Programs for Aid in Math

Developmental Variable	Math Aid		No Math Aid		F	df	p
	$\bar{X}$	SD	$\bar{X}$	SD			
Physical Development Subscale	1.98	.47	2.32	.53	4.81	1,37	.05
Ties Shoes	2.15	.75	2.68	.89	4.17	1,37	.05
Holds Crayons & Pencils Correctly	1.80	.52	2.32	.67	7.21	1,37	.01
Emotional-Social Dev. Subscale	1.91	.43	2.29	.37	8.87	1,37	.01
Emotional-Social Readiness	2.05	.60	2.53	.90	3.78	1,37	.05
Accepts School Routines	1.75	.44	2.42	.51	19.37	1,37	.001
Is Cheerful and Happy	1.65	.49	2.05	.71	4.33	1,37	.05
Accepts Correction	1.90	.45	2.32	.58	6.29	1,37	.05
Participates in Group Activity	1.85	.49	2.26	.56	6.01	1,37	.05
Observes Rules and Regulations	2.00	.65	2.47	.51	6.35	1,37	.05
Assumes Responsibility	1.80	.62	2.42	.77	7.80	1,37	.01
Mental Development							
Work Readiness	1.75	.64	2.32	.75	6.46	1,37	.05
Listens Attentively	2.00	.86	2.79	1.03	6.78	1,37	.05
Works Well Independently	1.90	.99	2.47	.96	4.16	1,37	.05
Total Progress Report	1.98	.48	2.31	.53	3.91	1,37	.05

Key: lower score = higher rating

Table 5

Means and Significant ANOVA's Comparing Developmental Variables for Students Who Had or Did Not Have Software Programs for Aid in Science

Developmental Variable	Science Aid		No Science Aid		F	df	p
	$\bar{X}$	SD	$\bar{X}$	SD			
Emotional-Social Development Subscale	1.65	.70	2.14	.39	4.91	1,37	.05
Emotional-Social Readiness	1.50	.58	2.37	.77	4.76	1,37	.05
Accepts School Routines	1.50	.58	2.14	.55	4.86	1,37	.05
Accepts Correction	1.50	.58	2.17	.51	6.00	1,37	.05
Observes Rules and Regulations	1.50	.58	2.31	.58	7.02	1,37	.05
Shows Consideration for Others	1.50	.58	2.06	.48	4.64	1,37	.05
Reading Readiness							
Prints First Name	1.25	.50	2.06	.80	3.82	1,37	.05
Sees Likeness and Difference	1.25	.50	2.09	.66	5.99	1,37	.05

Key: lower score = higher rating

STRATEGIES FOR SEQUENTIAL PHONICS INSTRUCTION USING COMPUTER SOFTWARE

Linda Mensing, M.S.Ed.
Anne Meyer, Ed.D.

CAST Resources, Inc.
76 Lafayette Street
Salem, MA 01970

Abstract

Teachers of learning disabled students often find it necessary to draw from a wide range of instructional techniques and materials when building an effective reading curriculum. The computer has recently been introduced into special needs classrooms, and many teachers have found it to be a motivational educational tool for their special needs students. However, in order to use the computer effectively, teachers must have access to software which supports their approach to phonics and reading instruction. This presentation will examine computer software programs useful for the remediation of reading disabilities. A carefully delineated progression of phonics skills will serve as the framework for examining a number of well designed programs.

Introduction

Systematic integration of the computer into the reading curriculum of special needs students requires matching the specific skills being taught to the appropriate software. Special needs students require "special" instruction precisely because general solutions do not work for them. Thus, teachers need to know which programs will work for their students and specifically what skills each program addresses. The following is an example of a development of skill sequence with software recommendations in early reading skills.

Teaching the Correspondence Between Spoken and Written Words.

In order to become proficient readers, children must be aware of the relationship between words that are spoken in our language and words that are written. With many preschool and kindergarten aged students it is important to introduce exercises which emphasize the structure of language (Camp, Winbury and Zinna 1981). Traditional activities appropriate for teaching this concept to young children include: having children learn to recognize their own name and names of other people who are important to them; observing labels on common objects and signs in their environment; and seeing pictures identified by the printed word representing them. Creating pictures, then labeling the pictures has proven effective for teaching the relationship between spoken words and written words, and there are a number of highly motivational software programs which have been used to effectively teach this concept.

Software Recommendations: Picture Perfect by MindPlay, Methods and Solutions, Inc., is a Mouse driven program which combines the ability to produce graphics, with a library of pre-drawn pictures and the ability to input text. In addition, the menu is picture based, making it possible for young children to manage the program somewhat independently. Using a language experience approach, the child identifies pictures he/she has produced on the screen and, with the help of the teacher, types in the labels for the pictures. The teacher can also type in a story developed by the child to describe the picture he/she has produced. Programs which might be substituted include: Kid Writer and Story Machine by Spinnaker software; Story Maker by Scholastic; Paint With Words by MECC (requires the Ufonic speech synthesizer).

Teaching Linguistic Awareness.

For learning disabled children who are slow to develop a conscious awareness of the linguistic components of speech, research shows it to be beneficial to provide linguistic awareness training (Conrad 1971, Ehri 1975). Such training should focus upon teaching children to listen to oral language, and discriminate words, syllables, and phonemes. Though phoneme awareness is necessary for the eventual task of associating sounds with their graphic representations, discrimination of phonemes can be difficult for some children. Because phonemes are overlapped in the spoken word, they are difficult to separate without some articulatory distortion. Thus, phoneme awareness activities should be used which focus the child's attention on sounds within words. Appropriate traditional activities might include: reciting nursery rhymes, playing rhyming games, and games which require the child to identify sounds in the initial, final and medial positions in words or syllables.

Software Recommendations. Exploration of sounds within words can also be done effectively using software programs such as Keytalk (uses an Echo speech synthesizer) by Peal Software; or Talking Text Writer (also uses the Echo) recently published by Scholastic. These talking word processing programs have been used effectively to allow children to explore sounds in words. When the teacher types in a word, the word is pronounced by the computer. New words can be made by changing a single letter in the word. One appropriate activity involves the teacher typing in a word, then having the children change the beginning, medial, or ending sounds of the word to make new words.

(Please note that for school aged learning disabled students, training in linguistic awareness does not preclude initiation of reading instruction. Linguistic awareness can very easily be combined with early phonics instruction.)

Teaching Letter Matching and Letter Identification.

Letter matching and letter identification skills can be taught quite effectively with the computer, because computers can provide drill and practice in a variety of highly motivational, multisensory formats for students who are having difficulty with memorization.

Software Recommendations. Appropriate software for Letter Matching includes Early Games for Young Children (possibly with an A to Z adapted keyboard); Early Learning I by Marble Systems; and Floppy Teaches Letter and Number Recognition by Floppy Enterprises. These last two programs offer teachers control of the letter symbols presented to the student, and the teacher can limit the number of letters presented.

It is also very effective to allow students to repeatedly experiment with StickyBear ABC's using the Unicorn Expanded Keyboard (in an A to Z layout if desired) and the Echo speech Synthesizer. The synthesizer says each of the letter names as the child touches them on the keyboard, thus providing the child with simultaneous auditory and visual stimuli. Neither the expanded keyboard nor the speech synthesizer are necessary if a teacher or parent is available to work with the child.

Teaching Sound-Symbol Association.

Once a child has developed an awareness of the phonemic structure of language and can label many of the letters of the alphabet, instruction in sound/symbol association is usually initiated. With instruction, children should learn to provide the sounds for visually presented letters, and should automatically recall the names of letters in response to orally presented sounds. Most remedial approaches provide specific techniques for teaching sound/symbol association (Engelmann 1969; Gillingham and Stillman 1956; Slingerland 1971). Thus is it important for teachers to be able to integrate computer software with the reading/phonics approach they are using. For example, some instructional systems present letters in a very specific order. Therefore, software which presents sound/symbol associations in random order, or requires the child to be able to identify nearly all of the letter sounds before being able to work independently with the program, may not be useful.

Software Recommendations. For practice discriminating and identifying single letter sounds in the initial position within words, Sound Ideas - Consonants by Houghton Mifflin has been useful. Each letter sound is presented individually and the program is very easy for students to work with independently. Because the teacher chooses the specific sound/symbol correspondence for each student to practice, this software can be used to supplement most beginning phonics programs. Though less versatile, the program First Letter Fun by MECC has proven highly motivational for drill and practice in sound/symbol association.

Practice with sound/symbol correspondence should also include activities such as deletion and insertion of sounds into words. Sound deletion and substitution activities with Keytalk might include deleting sounds from words to create new words. For example, if the letter S is deleted from SHEEP the word PEEP is created.

Teaching Mnemonic Devices.

Research with learning disabled students (Gillingham and Stillman 1956; Slingerland 1971; Gallistel, Fisher, and Blackburn 1977) suggests that it is very important for learning disabled students to memorize a standard "key word" for each sound symbol correspondence. The key word functions as a mnemonic device for sound recall and gives the child continued practice in identifying and isolating sounds within words. For example, when the letter S is present, the child might recite, S/sun/S. As new word sounds are introduced, the teacher provides daily review of previously taught sounds.

Software Recommendations. Experience with computer aided instruction has shown that when a teacher is available to help students identify picture and letter names and make appropriate sound/symbol correspondences, some students spontaneously begin to use words from StickyBear ABC's as key words to aid them with the recall of specific letter sounds. The program, First Letter Fun with its keyword chart at the end of the program can also be used for this purpose.

Teaching Two-Phoneme Syllables.

After students have learned sound/symbol correspondence for single letters, two-phoneme syllables should be introduced. Thus prior to reading CVC words, students should be taught to read short vowel-consonant clusters such as: AN, UP, AT (Camp, Winbury, and Zinna 1981).

During the initial stages of reading instruction, it is not unusual for a child to phonetically segment and spell a word accurately, yet be unable to read the word seconds later. In order to read words, more than segmenting skills and knowledge of sound/symbol correspondences are necessary. If the child uses a letter by letter decoding strategy, he/she frequently produces a nonsense word (bah/a/tah = BAT). In order to minimize the distortion problem inherent in blending sounds, students might be taught to cluster certain letters together.

Software Recommendations. SuperLead by Lead Associates provides for developmentally sequenced presentation of phonics, beginning with vowel-consonant (VC) syllables such as AM, AN, and AT. Throughout the Superlead program, students are taught to automatically recognize clusters of sounds.

Construct A Word by DLM has also been found to provide effective instruction in sound/symbol correspondence and sound clustering, starting with VC syllables.

Software designed to provide spelling practice, such as Magic Spells by Learning Company can be useful for giving students further experience with sound/symbol correspondence and sound clustering. Because this program has a management system, teachers determine the specific lesson content their students are to work with. Thus teachers can begin use of this program at an early level, for example, with VC and CVC words.

Language experience stories have been found to be a highly motivational method of providing practice with sound/symbol correspondence. Using a word processor such as Magic Slate which has a large 20 column (primary size print), students might be asked to type initial and final consonant sounds of words, familiar VC syllables and sight words, while the teacher assists by typing portions of unfamiliar words, spaces between words, and the appropriate punctuation.

Teaching Vowel Sounds.

Results of studies investigating children's decoding errors suggest that the greatest proportion of errors occur on the vowel sounds. Despite being taught to cluster sounds together when decoding, children often continue to have difficulty decoding vowels (Fower, Liberman, and Shankweiler 1977). Because the pattern of surrounding letters within a word determine the vowel sound within each syllable, numerous investigators have recommended that children be taught to observe the relationships between the consonants and vowels within words (Camp, Winbury, and Zinna 1971). Six basic word structures or syllable types have been identified (Gallistel, et al, 1977) and it has proven beneficial to teach these to students with reading disabilities.

Software Recommendations. SuperLead by Lead Associates provides instruction, as well as drill and practice using each of the six syllable types. The program presents them in sequence from easiest (closed syllables with short vowel sounds) to the most difficult (vowel-r syllables).

Additional practice with each of the six syllable types can be provided through use of the software Hint and Hunt and Construct A Word by DLM. These are both multiple disk programs which allow the teacher to assign specific types of vowel syllables for students to practice. These programs have proven very motivational and quite effective at helping students to automatically recognize various vowel syllables.

Teaching Automaticity.

The importance of speed drill and building automatic word recall has been recognized in numerous studies of reading comprehension (Gallistel et al, 1977; Camp, Winbury and Zinna 1981). By using speed drills, word recognition becomes more rapid, which will facilitate comprehension of sentences and paragraphs. In order to comprehend what one is reading, the recognition of each word must be rapid or automatic. If word recognition is slow and difficult, all the reader's attention is focused on the decoding process. Therefore little attention is paid to thinking about or understanding the text.

Software Recommendations. Software programs which provide well designed speed drills, while allowing some flexibility in the actual speed of performance required from the student include: Magic Spells by Learning Company and Word Wizards by MECC, Speed Reader and Spell It by Davidson Associates.

Teaching Word Meaning.

With learning disabled students teachers must also be careful to avoid misleading children into thinking that reading rapidly is the only thing that is important for good comprehension. Students must also be provided with many opportunities for reading for information.

Software Recommendations. At a single word level, reading for information and understanding can be encouraged with games such as Word Attack by Davidson, a vocabulary development program. The program, Crossword Magic might also be used to build understanding of single word meanings.

Teaching Early Reading Comprehension.

During the initial phase of instruction in reading comprehension, the student's attention should be focused on the content of the text. Thus the comprehension questions presented should demonstrate to the student that the information is located in the material they have just read. At this level the student would be asked to respond accurately to WHO, WHAT, WHEN, WHERE, HOW and WHY questions.

Software Recommendations. The reading comprehension program, Scuffy and Friends by Hartley Courseware provides authorable stories with comprehension questions and refers the student back to the text if a comprehension question is failed. Writer Rabbit by Learning Company has a motivational game requiring that students respond to WHO, WHAT, WHERE, WHEN, and WHY questions.

Gathering and remembering information while reading sentences and paragraphs can be encouraged with software programs such as: Dragon's Keep by Sierra On-Line and Where in the World is Carmen San Diego? by Broderbund.

The strategies present for integrating computer software into the phonics curriculum of learning disabled students do not constitute a complete instructional program for beginning reading, but it is hoped that the techniques can be integrated into a wide variety of initial reading approaches. Teachers should be able to selectively use any or all of these techniques to meet the particular needs of an individual reader or group.

References

Camp, L.W., Winburg, N., Zinna, D.R., 1981. Strategies for Reading Instruction. Baltimore, MD.: The Orton Dyslexia Society.

Conrad, R. 1971. The Chronology of the development of covert speech in children. Developmental Psychology 5:398-405.

Ehri, L.C. 1975. Word consciousness in readers and prereaders. Journal of Educational Psychology 67: 204-212.

Engelmann, S. 1969. Preventing Reading Failure in the Primary Grades Chicago: Science Research Associates, Inc.

Fowler, C.A., Liberman, I.Y., and Shankweiler, D. 1977. On interpreting the error pattern in beginning reading. Language and Speech 20: 162-175.

Gallistel, E., Fisher, P., and Blackburn, M. 1977. Sequence of Objectives for Teaching and Testing Reading Hamden, Connecticut: Montage Press.

Gillingham, A., and Stillman, B. 1956. Remedial Training for Children with Specific Disability in Reading, Spelling and Penmanship Cambridge, Mass.: Educator's Publishing Service.

Slingerland, B.H. 1971. A Multisensory Approach to Language Arts for Specific Language Disability Children: Guide for Primary Teachers Cambridge, Mass.: Educators Publishing Service.

SOCIAL STUDIES/APPLICATIONS

C. Jane Hydrick, Kay Gore
Joanne Troutner
James Ellis, Roger Bybee
Paula Panto, Mark Nelson

ABSTRACT: Primary Students' Uses of Data Bases as Tools for Lifelong Learning

C. Jane Hydrick, College of Education, Arizona State University and Kay Gore, Arizona State University

A close examination by the speakers of the research and literature revealed little, if any, use of data base software with schoolchildren in kindergarten through third grade. Typically, reports of computer use with children of this age group focus on the computer as a tool to build and/or reinforce basic skills. However, data base software offers children within this age group the same power for handling and sorting information as it does older children and adults.

Cognitive development theory provides a rationale for having children of primary age sort and process information -- the design and purpose of data bases. Piaget stresses that in the preoperational stage (approximately 2-6 years old) children are receptive to the concept of simple classification. Data bases can be a means of helping children learn and reinforce classification skills. Unlike some noncomputer activities that engage children in classification, the computer enables children to change the classification category easily. Not only do children manipulate the information within the classification, they discover the nature of the classifications themselves.

In the concrete operations stage (approximately 6-11 years old), Piaget hypothesizes that children are receptive to class inclusion and seriation. Again, data base software can be the children's tool. Here, children, in their development of data bases, learn that the information contained within their data bases is limited by their own knowledge as well as by their sources of information: The children are limited by the number of sources of information as well as the perceptions of those who authored the sources. Thus, they will learn that all information is perception- and error-bound. The data base, then, is regarded as the means, not the end. It becomes a tool limited only by the child's imagination and willingness to create questions rather than to seek answers. The importance of the data base is not simply as a source of information or an information retrieval system, but what the children themselves do with that information.

Support for the use of data bases with primary-age children is also found in the research of Bloom, Vygotsky, and others. The applications of data bases with young children should begin in the concrete world that they know if they are to realize such software as intellectual tools. In a project to examine the ability of primary-age children to use data bases effectively as educational tools, C. Jane Hydrick had a class of second-grade children create data bases on themselves, using such fields as favorite foods and hair color. The information that went into the data base was information that they were familiar with; there was no new or unknown information. In addition to forming new links of knowledge, the children formed new social links based on such attributes as favorite food and favorite color. The sense of class community grew too as they learned that they all had common bonds of information. The various sorts they printed out provided the class with a record of their new links.

Once the children realized the power of the data base as a tool and they felt comfortable enough with it, they could transfer that power to a content area where they had little or limited knowledge. They analyzed the information they gathered from different sources and then synthesized the information. This synthesization produced new perspectives and new information not only on as basic a level as field, but on the whole subject being studied. They made links within the subject and, ultimately, with other subjects, thus creating a sense of community with all knowledge.

Children who are aware that knowledge is dynamic and interacts with the knower are better prepared to be lifelong learners because they know that the power for learning resides within them and that the tool to exercise that power rests at their fingertips.

ABSTRACT: Study and Information Skills--Interactive Video Style

Joanne Troutner, Computer Education Chairperson, Tippecanoe School Corporation, Lafayette, Indiana

This year sixth-grade students at Klondike Middle School are learning information retrieval and handling skills in a new manner. These students are using interactive video lessons to receive and provide instruction and practice in retrieving information for reports, presentations, and long-term projects. The goal of the project is to determine if interactive video makes a measurable difference in students' information retrieval and processing skills or affects the students' attitudes toward information handling instruction. The venture involves using various interactive video lessons, which were prepared by the library media specialist as well as the science and social studies teachers. This team approach for developing lessons utilized the expertise of an instructional design person and a content specialist.

In order to obtain some data on the effectiveness of the interactive video lessons, a control group of sixth-grade students will receive similar instruction from the library media specialist instead of using the interactive units.

Information will be gathered from the students on their attitudes toward using interactive video lessons. In addition, a test of information retrieval and handling skills will be given to all sixth graders.

ABSTRACT: Appropriate Applications of Educational Computing for Elementary School Science and Health

James D. Ellis and Rodger W. Bybee, Biological Sciences Curriculum Study, 1115 N. Cascade Avenue, Colorado Springs, Colorado 80903

The Biological Sciences Curriculum Study (BSCS) and the International Business Machines Corporation (IBM) collaborated to study science and health education in the elementary school. The major focus of this study was to determine appropriate applications of educational computing for elementary school science and health. The major goals of the study were: (1) to design a framework for an elementary school science and health program consistent with current trends and needs as identified by the education and scientific communities; (2) to determine the appropriate uses of microcomputer technology; and (3) to produce a plan for implementing exemplary elementary school science and health materials including educational computing.

The study began with a major effort to gather the most recent information on learning theory, teaching, the school environment, state and local requirements, current curriculum, instructional practices, educational computing, and program implementation. Also, computer software for elementary school science and health was reviewed to identify examples of appropriate applications. Next, a scope and sequence of a proposed curriculum for elementary school science and health was developed. Based on that proposed curriculum, recommendations were made for applications of educational computing to enhance learning and teaching.

A major report, available to educational leaders, summarizes this study of science and health at the elementary level. The report contains a contemporary rationale and goals for K-6 science and health education and includes the knowledge base for decisions and recommendations for integrating educational computing into instruction for elementary school science and health.

ABSTRACT: Camp at the End of the Universe

Paula J. Panto, Computer Education Specialist, Museum of Science-ComputerPlace and Mark Nelson, Thayer Academy/Museum of Science

A new project, called Camp at the End of the Universe, was initiated at Boston's Museum of Science in the summer of 1986. This program involved 42 campers, ages 9-13 in all-day activities relating to science and computer discovery.

As part of the camp, ComputerPlace, the Museum's computer learning center, offered a 25-hour, 2-week program focusing on the theme of space, time and technology. In order to coordinate with activities designed by the Museum's Education Department, the computer curriculum focused on three areas: (1) Maxxon Rescue Mission; (2) Adventure Programs; and (3) Robotics.

Part I: The Maxxon Rescue Mission: As part of Starfleet Maxxon, each camper had a 3-day mission to help save 800 people from a planet. Each camper learned about word processing, data base and spreadsheets as ways to collect vital and accurate information. All campers received a printout of each activity.

Part II: Adventure Programs: Campers developed their problem solving skills by working in teams to solve space adventure programs.

Part III: Robotics: LEGO Logo introduced campers to the mechanics of robots, e.g., levers, gears and simple machines. This project allowed campers to design, build and test Lego structures. Campers used LEGO Logo and a computer to build and program creations such as washing machines, stop lights, ferris wheels and egg beaters.

Throughout the two-week period, campers designed and titled their own newsletter. As a finale to the camp, a slide presentation was shown and each camper was presented with specially tailored computer award certificates.

APPLICATIONS

Thomas R. Harris
Joseph R. Meadors
Judith Montgomery
Francis M. Betts III, Sarah Poncz

ABSTRACT: Using *Lotus 1-2-3* to Teach Information Processing Basics

Thomas R. Harris, 622 West 114th Street, #4A, New York, NY 10025

Computers have finally arrived in the nontechnical workplace and in schools outside of the science departments. Two major types of software packages have created the broad interest necessary for that occurrence: word processing and spreadsheet packages. *Lotus 1-2-3* is the most popular of the latter types; it has allowed managers in business and academia to process and present information using the power of a computer, but without programming. To prepare students for a world in which instantaneous information processing is the major application of computers, *Lotus 1-2-3* is an ideal teaching tool.

The author uses *Lotus 1-2-3* to introduce data base processing concepts to students at the high school and college level. As part of a co-op work study program, students have used Lotus to manipulate and present data related to a project in bibliographic information storage and retrieval. The Macro programing language, which runs within the Lotus spreadsheet, is similar to Basic and has allowed students to write simple software for information processing. In a separate, after-school program, the author presents a simple definition of a data base as an indexed body of information; gives examples of data bases in the home, such as telephone books, cookbooks, and catalogs; and then shows how these data bases can be computerized with *Lotus 1-2-3*.

ABSTRACT: Replacing Computer Literacy

Joseph R. Meadors, Brebeuf Preparatory School, 2801 West 86th Street, Indianapolis, IN

This one-semester course integrates the areas of keyboarding and computer literacy with real-world skill such as banking, credit, and contracts.

Students spend the first nine weeks learning the keyboard and proper keyboarding technique on manual typewriters. A unit on checking is interspersed to alleviate the tedium of learning to type. Within the checking unit, the student will learn the necessary processes involved in balancing one's personal checkbook. These processes include deposits, drafts, service charges, outstanding checks, and NSF charges. In the keyboarding area, a minimum competency of 15 w.p.m. is expected after nine weeks.

In the second nine weeks, students are given computer orientation and additional keyboarding practice. The student is introduced to elementary word processing concepts and should then be able to use the word processor as a tool toward his or her education. Mixed in this quarter are concepts of borrowing money and simple contracts. The borrowing money unit involves interest calculation, types of credit, and comparing credit. The contract unit covers how the law views contracts with minors and one's rights and obligations within a simple contract.

When the student finishes this semester, he/she should have a firm grasp of the following concepts:

1. Touch typing technique;
2. Checkbook operation;
3. Computer orientation;
4. Costs of borrowing money;
5. Elementary word processing;
6. Simple contracts.

ABSTRACT: Cryptographers, Thunderstorms, and Computer Games: A Computer-Based Approach to Functions and Graphing Linear Equations

Judith Montgomery, The Logo Algebra Project, Lesley College, Cambridge, MA

One problem with teaching students about graphing functions is the almost wholly abstract nature of most standard curriculum activities. We are designing a set of algebra curriculum materials that use the computer to introduce functions and graphing linear equations in a concrete and "real-life" context. In these units, students will first become cryptographers, who must learn to crack codes, and then weather analysts, who must determine the distance of approaching thunderstorms. In order to successfully carry out their tasks in each case, they must use functions and graph equations at their computerized workstations. In these activities we use the image of an input/output machine as a model for functions. As cryptographers, students must design function machines for encoding and decoding messages and implement the machines by programming them in Logo. To crack difficult codes, they must make a computerized graph based on the encoded key word and analyze its slope and intercept. As weather analysts, they first do calculations and make tables and are then introduced to the power of using a graph to replace repeated calculations. They then create and analyze graphs on the computer to determine the distance of approaching storms. In each case an efficient way to represent the work they are doing is to express it as an algebraic function. The power of this representation is pointed out as they work.

The experience with the tool use of functions and graphs is then used at a more abstract level to help them play a series of function and graphing computer games, where they match wits with the computer and can try to invent problems that will stump their fellow students.

We have been developing these and other experimental algebra curriculum materials under an NSF grant titled The Logo Algebra Project.

ABSTRACT: Building a Communications Network

Francis M. Betts III, Executive Director, Philadelphia Renaissance in Science and Mathematics (PRISM), Sarah Poncz, Mathematics Department Head, Roxborough High School, Philadelphia, PA

High Tech Talk is an electronic bulletin board intended as a means of linking mathematics and science teachers to resources in industry. HTT presently offers electronic mail, a synoptic calendar of events, a variety of text files on educational resources, 20 SIG bulletin boards, and upload and download capability. Discussion will focus on design considerations and critical issues in implementation, including hardware, software, training, and system support requirements. Barriers to implementation will be reviewed, and strategies for developing teacher involvement will be presented.

COMPUTER AIDS FOR THE TEACHING OF CRITICAL READING
IN THE
LIBERAL ARTS CURRICULUM

Joseph Orndorff
Department of Special Studies
College of Arts and Sciences
Duquesne University
Pittsburgh, PA

Abstract

This paper describes the use of the computer in the teaching of critical reading skills in the Concentrated Studies program at Duquesne University.

Critical reading ability lies at the foundation of post-secondary education; however, increasing numbers of students enter higher education without these skills. This puts a tremendous burden on the instructor to compensate for this lack in the students, and can often result in the de-emphasis of the role of reading in the college and university classroom.

Our goal is to use the computer to provide coaching and drill-and-practice support for students who are inadequate readers. This we try to accomplish by the use of innovative computer based coaching programs as a supplement to classroom education. Our package, called LECTOR, consists of three programs, available on diskette, which can be used with the IBM PC. The first program, called ABSTRACT, leads students through the inspectional reading of a book, and also allows for prompted summary writing. The second, called INTERLOCUTOR, is a simple authoring program, which can be used to gloss a machine-readable text with annotations and interactive questions. This program also features a dictionary function, which allows the user to call up definitions of difficult words in the text. The last program, SCHOLION, is an editor, which provides a flexible environment for the dynamic glossing, footnoting, and annotating of text.

Our approach to critical reading, and our use of the LECTOR programs in the classroom is also described.

A major skill that many post-secondary students lack is that of critical reading: the ability to read difficult and challenging material and understand what they are reading. While these students may be well-intentioned, and hard workers, often they are unable to pick out main ideas, important sentences, etc., and see how these are combined by the author into a coherent and sustained argument.

Inability of students to read well, and understand what they read may be the single most important academic problem in post-secondary education. The *Chronicle of Higher Education*, pointing out the "inability of a majority of high-school students to understand, summarize, and explain complex materials," predicts that these students "will have difficulty coping with the demands of higher education." (September 25, 1985, p. 32) And, as is obvious to anyone involved in higher education, this prediction shows no sign of proving to be false or premature.

The seriousness of this problem is well-documented, of course. For example, *The Reading Report Card*, September 1985, found that only five percent of 17-year-olds possessed advanced reading skills. What is more troublesome than the low percentage of highly-skilled readers, however, is the fact that the reading samples used with the students in the study were carefully selected/constructed texts, which may, in fact, bear little resemblance to the kinds of reading actually required of post-secondary students. Accordingly, even if the incoming college student is among the few who possess "advanced" reading skills, will this student be able to read and understand Plato or Aristotle, Shakespeare or Faulkner, the *Constitution* or the *Declaration of Independence*?

By the time students reach post-secondary education, they have certainly achieved some level of reading ability. This level is not equal for all students, however, and is one of the things that serves to differentiate the student population. In general, it seems clear that the better students are the better readers, who have efficient guidelines for approaching written material, but the difficulty is that often even the best readers are still not equal to the task of reading difficult material, and show the most astonishing informational gaps when it comes to reading. It is not unusual to have students who do not know what the function of a footnote is, what the table of contents does, and why a book may have an index. When asked to state the subject matter of a work, the student may be at a complete loss; any written work, of whatever length is often referred to as a "story." What is perhaps even more astonishing is that these problems can be seen to persist often to the graduate level.

Often, students perceive their lack of reading ability and wish something could be done to help them read better. With the best of intentions, students may endlessly pass their eyes over texts, with little or no effect. They may spend hour after hour in frustrated boredom, puzzling over an apparently impenetrable piece of writing, before foundering and giving up completely. Instructors, frustrated themselves, may try a number of approaches in an effort to meet this problem, but increasingly and inevitably, the reading load is lessened. Instructors may merely tell their classes about great works rather than have students actually read them, or resort to predigested versions of texts, which are carefully edited and shortened in the name of "readability." Thus, inability of students to read effectively brings about what H. George Hahn II, writing in the *Chronicle of Higher Education* (Oct. 30, 1985, p. 84), calls an "Academic Gresham's Law," as watered-down textbooks and ever-lighter reading loads replace the reading of challenging basic works. This both increases the reading problem and erodes our cultural heritage, as more and more of the written tradition of the West slips away from the classroom. The result is that, inevitably, we continue to become a "culture without culture," at least insofar as a written tradition is concerned.

One reason for the reading problems at the higher educational levels is that reading is usually not taught after the first few years of schooling, except in a remedial manner for the nearly illiterate, or for those students with developmental problems of various kinds. The teaching of CRITICAL reading skills, the ability to be an active and discriminating reader, capable of reading difficult texts well, is virtually ignored. The result is that even the best post-secondary students are often employing the same reading strategies that they may have used in grammar school. If, in fact, these students have improved their reading strategies, it is often not as a result of having been taught new approaches to printed material, but rather due to a very haphazard personal reading. This kind of personal development is obviously not a bad thing, and should be applauded and encouraged. Often, however, this self-development does not occur, and the student, out of frustration with his/her ability to read only the least challenging materials, comes to feel that reading is a boring, difficult chore, which is best avoided, and done only under the greatest duress, if done at all.

Critical reading is an intellectual skill, and as such, it is learned by doing. There are no formulae to memorize, no mystic techniques that will suddenly improve reading ability. People who are good readers are, by and large, people who read a great deal, this reading allowing them to internalize rules and strategies of good reading, which they apply unconsciously. As a byproduct of developing good reading skills, the reader can also be seen to develop some understanding of good writing. For example, someone who has read widely well may have an intuitive feel for grammatical rules without having explicit knowledge of those rules -- bad grammar just "doesn't sound right."

It can be observed that the strategies of experienced readers involve such things as asking questions as they read, noting interesting features of the text, drawing on their experience as a reader, and the like. These techniques are then articulated and combined, often unconsciously, into what we call critical reading ability. The greater this ability, the more capable readers are of interacting with any text they choose.

There is a good test for this. Place a book in the hands of a practiced reader, and almost unconsciously he/she will begin systematically inspecting the book, noting the title, the author, whether it has an index and table of contents; he/she will begin classifying the book, mentally placing it in a number of categories, drawing on his/her experience with other books; he/she may even begin leafing through the book, skimming to determine the author's purpose, argument, etc. Give the same book to a nonreader, and if he/she interacts with it at all, which is more than somewhat unlikely, he/she will do so only in an uncoordinated and disjointed manner. At the end of an hour, the good reader will be able to tell you a great deal about the book; the poor reader and the nonreader will be unable to do so.

In general, as we said, one becomes a better reader by reading. But, given the relative lateness in the academic careers of post-secondary students, combined with their need to develop reading skill quickly, there is little time to waste. The students are simply not in a position to take several years off from their schooling to read challenging material and so improve their reading ability by the self-formulation and internalization of reading rules. They need guided practice, which can help them read more efficiently by pointing out the things they should look for as they read. In effect, the students need an experienced reader who can sit down with them and give them personal coaching, pointing out what they should be seeing in the text. Ideally, this coach would read the book with them, asking the student the kinds of questions that critical readers ask themselves as they read. If this coaching is done long enough, the student can possibly learn how to imitate their coach and do this kind of reading for themselves.

We also feel that the most effective way to practice reading and thinking skills involves using materials that our tradition tells us are the best with which the human mind can occupy itself. Call these "classics," "great books," "masterpieces;" whatever you prefer; these are the works thought to be the greatest our culture has produced: works such as the plays of Aeschylus, *Moby Dick, The Canterbury Tales, the Bible, Crime and Punishment,* and so on. But, given that these are the best, the most challenging, the most thought-provoking, they are also the most difficult. This is not because they are badly written, but rather because these works have so much to offer. To a student who has reading difficulties, such texts, though long the basis of the liberal arts curriculum, could prove overwhelming.

Dealing with problems in education is not new to Duquesne University, which has a declared mission to the community to reach out to students often ignored or overlooked. Our Concentrated Studies program, for example, has been set up to deal exclusively with incoming students who lack academic expertise, but not ability, and who would not ordinarily be admitted to higher education at our university. These students are often quite bright, but for one reason or another, often do not know HOW to do well in an educational setting. In working with the students in this program, we have seen the importance of READING ability as a BASIS for developing all the other intellectual skills needed in post-secondary education. This perception of the importance of reading has also been confirmed by experience with students outside the Concentrated Studies program. We have seen that reading difficulties cut across the undergraduate spectrum; even the students who are among the best and the brightest may be inadequate readers. But, the students in the Concentrated Studies program are a population that shows consistent and obvious reading difficulties, and so, these students, as a group, have a pressing need to develop a high level of reading skill as quickly as is possible.

In wanting to help our students become critical readers, we want to help them develop the ability to do a number of things:

- Inspect a book. This would involve noting basic information about the book, classifying it according to subject matter and type, and noting the structure of the work.

- Read and interact with a work, including answering questions about the author's strategies, and the content of the work.
- Write a summary of the work.
- Develop and articulate criticisms of the work.
- Use major texts in the western tradition as materials on which to practice, thereby providing the students with challenging materials, as well as a better understanding of the western intellectual tradition.

Given that the students have a need for intensive coaching and drill-and-practice beyond the ability of the instructor to satisfy, it would seem that the computer could provide an environment within which our students could acquire critical reading skills. Unfortunately, since there seemed to be no pre-existing software that could provide us with the type of environment we sought, we decided to develop our own software for the teaching of critical reading skills.

Our reading software package, called LECTOR, consists of three programs, ABSTRACT, INTERLOCUTOR, and SCHOLION, designed to run on the standard and ubiquitous IBM PC. These programs are written in TURBO Pascal, and are small enough so that the programs and their data files will fit handily on floppy diskettes.

The ABSTRACT program is designed to lead students through the study of the structure and basic content of literary works. The user is first asked to give the title of the work, name of the author, date of the work's writing and publication, etc. After this, the student is given the opportunity to classify the work according to subject matter and literary type. There is a built-in dictionary support for this task, since students are often at a complete loss as to what a novel is, or a play, or a short story, or how the subject matter of a philosophic work differs from a piece of imaginative writing. This dictionary capability is completely flexible: the instructor creates the dictionary file that the program uses. The next task is for the user to schematize the structure of the work in response to machine-generated questions. Finally, the program allows for prompted summary writing and asks for the student's critical opinion of the work. Natural exit points occur in the program at each shift to another task. Student responses are written to a work diskette. No provision has been made yet for machine checking of some of the responses for reasons of space on the diskette, and also to allow for maximum flexibility.

INTERLOCUTOR is a flexible tutorial authoring program, which features various kinds of question and answer and screen manipulations, windowing, and the ability to have an on-line dictionary of words and concepts, again created by the instructor. This program is designed to allow instructors to lead students through the critical reading of text by means of a machine-based dialectic, though it is also useful for generating standard Q & A tutorials. There is no capability for machine scoring at this time, since our students indicate that the thought of being scored by the machine makes them uncomfortable.

INTERLOCUTOR is typically used by us to create lessons in which text is presented in one window on the screen, and glosses on this text (questions and observations) in another. Questions can be presented either singly, or in sets, with many variations: ones with hints to help the student when an answer is wrong; ones that simply explain why the student is wrong, and then move on; ones that give more information, whether the student is right or wrong; ones that ask rhetorical questions for the student to think about; ones that take short typed-in answers, which the program then parses and checks. Questions may have a number of possible answers and may be presented in sets such that one question builds on another, or such that questions are asked from the set at random. In addition, the text being presented may also be enhanced by underlining, highlighting, reverse video, blinking characters; boxes can be drawn on-screen; text may be manipulated in a variety of ways; simple diagrams may be inserted, etc.

The third program is the LECTOR package, called SCHOLION, presents a powerful editing environment designed expressly for text glossing. Using SCHOLION, text may be footnoted; interlinear markings are possible; categories may be created and used to label themes and concepts in the work; and a variety of screen manipulations are possible. In addition, all of the glosses and manipulations can be called up, consulted, and dismissed as the user finds necessary. The result is a powerful, flexible, and learner-centered environment for the presentation of an interaction with textual material.

While SCHOLION was designed to allow for the glossing of sonnets with commentary and analytical markings and observations, its uses can go far beyond this. One use, for example, is the compilation of a data base of footnotes and commentary on a text. Another is the intensive glossing of a short work, including related readings, translations, critical commentary, and the like, all to be called up, examined, and dismissed as the user likes, within the SCHOLION editing environment.

INTERLOCUTOR and SCHOLION are designed to be used with machine-readable texts. Ordinarily, this would involve much tedious typing of material, but we have the limited capability to create machine-readable texts with a fraction of the effort. We have used a university-owned Kurzweil Reading machine to optically scan text and read it over to an IBM PC. (A use of this Kurzweil product, which is original to us, and which has been publicized by the Kurzweil company.) We now have a small library of more than 50 texts on diskette, including works by Milton, Bacon, Mill, Emerson, Aquinas, and others. All of these texts are in English translation, though the making of foreign language texts is also contemplated. Of course, we use materials that are in the public domain, for the most part, so as to avoid any copyright problem.

With these texts, it is then possible to use the LECTOR programs to create interactive tutorials. In addition, having texts on diskette is a cheap alternative to buying books, the price of which is growing prohibitive -- students can purchase "hard copy" generated by a high-quality printer from the stored text. We also hope that these texts will be of interest to scholars, who can use them in conjunction with a number of currently available analytical programs. We are also presently engaged in creating a set of analytical programs, which will allow us to search, concordize, index, collate, and otherwise manipulate these texts in a variety of ways useful in literary studies.

The most important use of these texts and the programs remains the teaching of critical reading skills, however. Since the Concentrated Studies students at Duquesne have the greatest need to develop these skills, it is these students who have been the first users of the LECTOR programs. A two-course sequence, Critical Reading I and II, has been designed to give these students intensive practice in reading, and the programs are used to provide coaching and practice outside the classroom.

In the first course, Critical Reading I, subtitled "Argument Analysis," students are introduced to critical reading by way of grammatical and logical analysis. The focus of the course is political philosophy, specifically the steps that led to the *Constitution of the United States*.

The reading list includes Aristotle's *Politics*, books I, III, IV; Machiavelli's *Prince;* Thomas Aquinas' *Treatise on Law*, QQ. 90-97; Locke's second essay on government, *Of Civil Government;* Montesquieu's *Spirit of Laws*, books I-XIII; Rousseau's *Social Contract;* the *Declaration of Independence;* the *Articles of Confederation;* the *Constitution of the United States;* and *The Federalist Papers*. All of these texts, except for the *Federalist Papers*, are in machine-readable form, and hard copy is generated by laser printer and purchased by the student instead of books. This hard copy is drilled with holes so that it will fit in a binder, giving everyone in the class what amounts to a looseleaf book. This format has many advantages, not the least of which is the ability to interleave notes between the pages.

The course, which meets three times per week, begins with a review of grammar, and a reading of Aristotle's *Politics*. The movement is from smaller units to larger: from word to sentence to paragraph to chapter to book. Along the way, grammatical structures are treated, including the use of modifiers, parallelism, subordination, and the like. After the grammatical review, Aristotelian logic is taken up: signs, words, and terms; division, definition. The exam for this section involves a grammatical and logical analysis of a section from *Politics*. Further, the students are required to write a summary of the reading as well as three short (1 page) essays on some aspects of the reading. Computer support is provided in the form of tutorials and glossed sections of texts, using the INTERLOCUTOR program. We have found that these lessons must be kept short, 30 to 60 minutes long, since the computer screen is not the best way to present written material. Further, the information on the machine is merely a fraction of the length of the whole -- students are expected to get machine coaching, then go on to read the text for themselves.

The next reading used is Machiavelli, and the focus of the class is on the logic of propositions. The exam, again, is an analysis of a section of this text. Again, the students write a summary of text, and are given three short essay questions that deal with the material.

The Locke reading follows, and the class focus shifts to paraphrasing and the nature and classification of paragraphs. Again, there are short, content-related essays to write, as well as a summary, section-by-section, of the Locke reading. Movement is from the paragraph to the chapter and book. The study of paragraphs, chapters, and books continues with Rousseau's *Social Contract*. The exam for this section involves an analysis of a section of Rousseau.

In addition to the above, there are frequent quizzes and pop-quizzes. Outside the class, the students are expected to do much reading and writing, while class time is devoted primarily to a combination of dialectic and lecture. In addition, as relevant to the reading, time is devoted to such things as the use of footnotes; the logical and grammatical strategies of the author; the construction of the text; the purpose of a table of contents, preface, introduction, etc.

By this time, the students typically begin to comment on the noticeable changes in their reading habits, as well as the increased attention they now pay to what they read. This change is especially noticeable if they are taking other reading-related classes.

At this point, about mid-semester, the emphasis of the class shifts to formal logic. The students begin reading the text of Aquinas' *Treatise on Law* for arguments, which they pick out, paraphrase, and diagram. Again, as in the previous sections of the class, INTERLOCUTOR tutorials are provided to give them coaching in picking out arguments, as well as other features of the text: examples, divisions, parallel structures, etc. Following the finding of arguments, syllogisms are touched on in a limited way: the basic syllogism in Barbara, compound syllogisms, abbreviated and expanded syllogisms. The whole point is to focus the student's attention on arguments, and get them to pay closer attention to the form and structure of philosophic exposition. The exam for this section involves reading a section of texts and diagramming the arguments found therein.

After this, the class moves on to a very limited coverage of first-order symbolic logic, which the students use in the symbolization of passages from the Montesquieu reading. This is useful because the students are essentially translating passages from the English text into another, very restricted language of variables and symbolized operations. As anyone who has done translating from one language to another knows, restating something in another language requires a high level of understanding of the text. In addition to the INTERLOCUTOR program, this section of the class also makes use of Preston Covey's ANALYTICS symbolic logic package. The exam for this section, obviously, involves reading and symbolizing a section of text.

Finally, the last few weeks of the class are spent comparing passages in books with one another, using the *Federalist Papers* and the American State papers as the central reading. Some of the propositions set out in the *Constitution* and the *Declaration of Independence* are isolated, and traced back through the class readings, in an effort to understand something about the development of thinking concerning those propositions and the dialectic that has occurred among the authors in question. The final is comprehensive and involves applying the skills learned throughout the semester to the careful reading of a section of text.

The goals of Argument Analysis are (a) to help the students develop skill in the critical reading of the text, and (b) to give them some background in a group of texts that are fundamental to the American political tradition. There has, as yet, been little chance for the student to do independent thinking about these works. This is not a bad thing, however -- if they cannot read and understand the works, it is doubthful whether they are really in a position to have any kind of critical thoughts at all concerning them. Thus, the emphasis in the Argument Analysis class is on what the authors say, and not what the reader thinks about what is said.

Giving students a chance to state their opinions concerning what they have read is the purpose of Critical Reading II, a Great Books Seminar, which also meets three times a week. In this class, which functions primarily by dialectic, students are given practice in thinking and arguing about what they read. Computer support for this class includes an ABSTRACT-guided inspection of the texts read, INTERLOCUTOR lessons on aspects of the reading, and ABSTRACT-guided summary writing. Every few weeks, students are given unfamiliar books, selected at random, on which they must perform an in-class, non-computer assisted inspectional reading. Written work, in addition to the summary and inspectional reading, also includes a short (1 page) journal entry. Other than this, the class proceeds solely by Socratic question and answer and discussion.

Readings for the Great Books Seminar include selections from all the readings used in the Argument Analysis class, as well as other texts: the *Epic of Gilgamesh* and *Genesis*, other selections from the *Old Testament*, a Greek drama, several Platonic dialogues, a play by Shakespeare, and others. One work or section of a work is read per week, with the goal not being a full treatment of the work, but rather an introduction to how to read these different works and the achievement of a certain familiarity with these writings as some of the most important our culture has produced.

Use of the computer for instructional purposes is still very new at Duquesne University, so there is a great deal of reluctance on the part of faculty regarding our approach. Happily, this is not the case with our students, who tend to be enthusiastic. It does not take long for our students to see the value of the computer lessons, and it is not unusual for them to begin asking why their other classes don't make much use of such lessons. Full-scale deployment of the LECTOR programs as course support for all the students in Concentrated Studies is now underway, and plans are being made for the use of the INTERLOCUTOR program in the university modern and classical language classes. Further, our software and approach have also been demonstrated to members of the Benedum Regional Consortium and at the EDUCOM conference. We are also open to inquiries and requests for demonstrations and are willing to make copies of our software available for Beta-testing at other institutions.

COMPARISON OF COMPUTER-BASED AND TRADITIONAL PRACTICE EXERCISES FOR DEVELOPMENT OF CONTENT AREA VOCABULARY AT THE COLLEGE LEVEL

Ernest Balajthy
State University of New York at Geneseo
Geneseo, New York 14454

Linda Bacon
Caledonia Elementary School, Caledonia, New York

Patricia Hasby
Letchworth Central School, Gainesville, New York

ABSTRACT

60 college level developmental reading students were each given three types of content area vocabulary lessons. One lesson type was based on worksheet paper and pencil tasks. A second type was delivered in the form of a computer-based video game. A third type was delivered in the form of a computer-based text game. Evaluation was carried out by a multiple choice achievement posttests and by student rating sheets. Findings indicated superior performance on the worksheet task and lowest performance on the video game task ($p < .05$). Time-on-task was shortest for the worksheet and longest for the text game. Student ratings of effectiveness did not correspond to actual achievement. The video game was rated as most effective and the worksheet was least. The two computer-based games were rated as more interesting than the worksheets.

Computer-assisted instruction is used frequently by both educators and students to develop and improve vocabulary (Geoffrion & Geoffrion, 1983; Balajthy, 1986). Little research, however, has been done to investigate effectiveness of computer-based instruction in vocabulary.

Alfano (1985) compared students' achievement in vocabulary using two different modes of instruction: Teacher-directed instruction and computer-based instruction. Alfano used subjects from a middle class suburban-urban area seventh grade with one- to three-year vocabulary deficits. The subjects were split into two sample groups. Sample group A was given vocabulary instruction using only a computer program. Sample B was given the same lessons under teacher direction using a print-out of the instructional material from the computer. Each group worked twenty minutes per day on the assigned lessons. The computer provided feedback on the work of Group A and the teacher corrected the errors with the class in Group B.

This study resulted in no prominent differences between the two methods of instruction. There was a slight mean difference in favor of the teacher-directed group. Alfano conjectured that this slight preference was due to the students' increased motivation from interaction with the teacher instead of a computer screen.

Rupely and Chevrette (1983) have listed the following reasons as to why students in a computer-assisted instruction program should, in theory, obtain higher achievement scores in reading than those using a traditional approaches:

1. Computer-assisted instruction programs are carefully designed to give individual instruction with immediate feedback.
2. Students are motivated through success that leads to the development of a more positive attitude and a more pleasurable learning experience.
3. Students are actively engaged in the learning task for the duration of the lesson.
4. Computer-based instruction can result in more time on task.

The present study was designed to investigate issues involved in computer-based vocabulary instruction for college students in a biology course. The issue of the relative merits of video game vocabulary softward, as opposed to traditional worksheets and text-based software, was of special concern. Video game instructional software offers many advantages for motivation, but most such software has serious disadvantages, as well (Balajthy, 1986).

Description

Subjects

60 freshmen, all enrolled in a college level developmental reading course due to poor performance on the Scholastic Aptitude Test, participated in the study. Each received credit toward completion of the course for participation.

These students were, for the most part, minority group students from upstate New York inner cities. Each had been identified by the college as being "high risk." Most were in the college's Educational Opportunity Program. All but three of the students were enrolled in a freshman biology course. The biology textbook was used as the source for vocabulary words used in the present study.

Experimental Procedure

Students in three classes were randomly divided into three equal groups per class. Each group completed each of the following:

Ditto vocabulary drill, with self-answer key

Vocabulary text drill grame (Quizit, published by Regents/ALA, Two Park Avenue, New York, NY)

Vocabulary video drill game (Word Attack!, published by Davidson & Associates, 6069 Groveoak Place, Suite 12, Rancho Palos Verdes, CA)

The exercise types were administered in counterbalanced time order to students to account for any disparities due to administration sequencing. That is, while all students in a class studied the same words on any given day, one-third of the students would be using the ditto exercise, one-third the computer-based text exercise, and one-third the computer-based video exercise.

Each lesson consisted of fiteen vocabulary words drawn from the students' biology textbooks. Students completed the drills in class, one per session. They immediately took a multiple choice vocabulary posttest after completing each drill.

Results

Results were analyzed using one analysis of variance, for immediate posttest achievement. The analysis compared performance on the three forms of instruction. In addition, three tabulations were performed, comparing the three forms of instruction on mean time-on-task per lesson, student rating of instructional effectiveness, and student rating of instructional interest. Mean scores are reported in Table 1.

In the analysis of posttest achievement, the main effect for instructional technique was significant at $p < .001$. Students using the worksheets achieved a mean score of 90.47. Mean score for the Quizit program was 80.70. Mean score for the Word Attack! program was 85.58. Scheffe' post hoc comparisons indicated a significant ($p < .05$) difference between the top score (for the worksheets) and the lowest score (for Quizit).

In the tabulation of time-on-task per lesson, students using Quizit spent the longest amount of time per lesson studying (mean of 44.42 minutes). The Word Attack! program required the second longest time (38.22 minutes). The worksheets took the least time to complete (31.74 minutes).

In the tabulation of student ratings of effectiveness, differences were rather minimal between methods. Students rated the Quizit program as most effective (mean score of 5.74), the Word Attack! second (5.50), and the worksheets third (5.00).

In the tabulation of student ratings of interest value, differences between the two computer programs were minimal. Word Attack! was rated at 5.75 and Quizit at 5.72. The worksheets' interest value was rated considerably lower, at 4.53.

Discussion

As with most research studies, findings raise more questions than they answer and point to the need for further study of the issues involved. Several interesting conclusions can be drawn, however, that are of concern to teachers evaluating vocabulary curricular needs and the roles that microcomputers can play in filling those needs.

Achievement

The finding that neither video game nor text-based software formats resulted in achievement superior to worksheet lessons was not surprising. It supports the notion that educators cannot look on computer drill and practice software as the answer to all educational problems, not even for problems involving fairly low-level instruction such as vocabulary drills. The authors suggest several reasons as to successful student performance on the worksheets:

1. The worksheets were self-scoring, providing rapid feedback as students looked at the answer keys, usually after completion of each exercise. The immediate feedback from computer-based instruction apparently was not sufficiently advantageous to overcome the benefits of rapid feedback.

2. The computer software programs were designed to enhance active student involvement by use of either video game action or a text guessing game. It may be that students involved in the study needed no such extrinsic means of promoting involvement with the vocabulary learning task. Existing motivation available for each type of lesson including knowledge that vocabulary words learned would aid in content area coursework and that grades received on the posttest would count toward a final grade in the reading/study skills course.

3. The involvement in the computer software tasks may actually have been largely off-task behavior. That is, in the video game, a great deal of concentration must be devoted to the game task of firing projectiles. In the text game, attention must be paid to the guessing task. The worksheets may allow students to devote full attention to the vocabulary learning task.

4. In addition, the computer tasks required that some of the students' attention be focused on actual operation of the computer, as opposed to vocabulary learning. This may have worked as a distraction, decreasing amount of vocabulary learning. Such off-task attention requirements may decrease as students become more familiar with software operation.

5. A final conjecture involves student metacognitive awareness of learning success or failure. Students using worksheets, a learning tool with which they are all familiar, may be better able to evaluate their own learning. They know when they have learned, so that their decisions to stop drillwork and take a test are better informed than when they are using unfamiliar computer software.

Time-on-Task

Not only did students fare poorest in achievement when using the computer software, but they also took the longest amount of time when on the microcomputers. Students using the worksheets displayed a no-nonsense, "Let's get the work done and get it over with" attitude. They worked rapidly and stayed on task. Students using the software were intrigued by the program operation and by the game qualities. Some were clearly motivated to spend more time using the programs. As noted above, however, this greater time-on-task did not necessarily lead to increased achievement.

The Quizit text game required the most amount of time to complete. Movement through the quiz was slow, demanding a good deal of time for each response. Rate of response was much quicker for the Word Attack! quiz. This pace of drillwork was apparently the reason for the greater amounts of time spent with Quizit.

Rating of Effectiveness

Students were in general unable to accurately rate the comparative effectiveness of the three learning methods. The most effective method , in fact, was rated the lowest by students. This demonstrates a metacognitive inability on the part of the students to recognize effective learning strategies. Such a finding is in general accord with research on underprepared college students and suggests that strong teacher direction may lead to increased success. The finding is also in accord with Garhart and Hannafin's (1986) findings that computer-based instruction incorporating learner-control often obtains less positive results that instruction in which the computer controls pace, time-on-task, and format.

Rating of Interest

Student interest in the worksheets was clearly lower than in the computer programs. Students apparently did not prefer the video format to the text game format in the software. While in fact some students were quite enthusiastic about the video game, some of this enthusiasm wore off even during the short duration of one lesson. The text game was based on the general format of the television show Name That Tune, and generated interest despite its lack of graphics.

It was not the intent of this study to measure whether increased interest on the part of students would lead to increased time-on-task in the long run. While in fact students did spend more time on the computer lessons than on the worksheets, this appeared to the researchers to result more from the programs' running time than from extra time spent studying voluntarily on the part of the students.

It must also be noted that students performed worst on those methods that they rated the highest. This was true both for students' rating of interest and for their rating of effectiveness.

Conclusions

The most solid conclusion to be drawn from this study is that, for drill and practice instruction in vocabulary, teachers should not indiscriminately discard traditional workbook-style drills in favor of technologically more sophisticated computer-based instruction. Instead, teachers must carefully evaluate a wide variety of factors that influence the success of such instruction.

Of perhaps most crucial concern must be a close consideration of motivational factors involved in instruction. It may well be that students will not benefit from the increased interest value of computer programs. While of course we do want to increase student enthusiasm for schooling and learning, it may be that other means of increasing motivation will achieve greater success than replacing dull skillbooks with video games. In the present study, for example, the motivation involved in doing well with content area and reading/study skill coursework appeared to overshadow the motivation provided by interesting game-like activities.

REFERENCES

Alfano, J. L. (1985). Seventh grade computer instruction vs. classroom instruction. Unpublished thesis, Kean College of New Jersey, Union, New Jersey.

Balajthy, E. (1986). Microcomputers in reading and language arts. Englewood Cliffs, NJ: Prentice-Hall.

Garhart, C., & Hannifin, M. (1986). The accuracy of cognitive monitoring during computer-based instruction. Journal of Computer-Based Instruction, 13, 88-93.

Geoffrion, L. D., & Geoffrion, O. P. (1983). Computers and reading instruction. Reading, MA: Addison-Wesley.

Rupley, W. H., & Chevrette, P. (1983). Computer-assisted reading instruction: A promising tool for enhancing teacher effectiveness. Reading World, 22, 236-240.

Table 1. Posttest and Survey Results.

	Worksheet	Quizit	Word Attack!
Achievement (maximum - 100%)	90.47	80.70	85.58
Time-on-task (minutes)	31.74	44.42	38.22
Effectiveness Rating (maximum = 7)	5.00	5.74	5.50
Interest Rating (maximum = 7)	4.53	5.72	5.75

DESKTOP PUBLISHING: AN INTRODUCTION

Frank Kulesa

Northern Illinois University

Abstract

This paper introduces electronic desktop publishing, which is currently being taught on the undergraduate level in the School of Art at Northern Illinois University. It is mainly written for educators who are interested in, but do not have any knowledge of desktop publishing. I hope it will serve as a beginning, showing why desktop publishing is desirable within a graphic design environment. This paper will mention the basic software and hardware required for an integrated desktop publishing system. Also projects and student reactions to working with this new design medium are covered.

Introduction

Electronic page layout borrows many principles from computer aided design and traditional drafting, enabling a graphic designer or layout artist to combine text, art and halftone photographs into a finished document.

Due to the wide use of the microcomputer in business and education, dramatic improvements in processing speed and memory space, desktop publishing is becoming popular. Although desktop publishing programs are not as sophisticated as workstation and dedicated publishing systems, several do perform a wide range of page design functions for printing tabloids, newsletters, small magazines, company reports, or any type of document a company feels is appropriate.

Most page layout manufactures have decided to support the IBM and the Apple Macintosh microcomputer. Although the IBM microcomputer is more widely used, its graphic capability has to be improved by a third party. Unfortunately these third party systems have no "standard" product between them. While the Macintosh does not have the calculating power of the later IBM AT and compatible microcomputers, it has powerful graphic capabilities. Software supporters have taken advantage of Macintosh's friendly interface that allows for easy manipulation of text and graphic files, developing a family of products with few compatibility problems. This with its lower cost has made the Macintosh computer a favorite with small design houses, free lance designers, and university graphic art departments.

IBM and Macintosh page layout programs require the text and graphic files to be produced on separate word processing, "paint" and "draw" programs. Art can also be generated from digitizers and scanners. By opening and closing these files while in the page layout program, the text and graphics are positioned together to form a finished document. Some programs will output what appears on the screen, while others show a mockup with block graphics and text. Probably the most important feature is the ability to interactively design and edit the document on the screen, which radically reduces production time. Changes on one page will automatically update parts of the rest of the document. After the document is completed, the information can be sent to a number of printing devices. These devices range from dot matrix printers for "proofs" to a laser printer or professional typesetter for a camera-ready hardcopy, or as the final output. For those that do not want to invest in output devices, some programs allow the document to be sent by modem or disk to a printer or typesetter.

The best software mimics the traditional drafting table of the layout artist. Text and graphics can be placed directly on the page or stored off to the side for later use. All layout tasks are handled directly on the screen, this improves the ease and quality of the layout. This eliminates a large part of the layout process and gives more flexibility in designing a document. Ideally the screen should be as large or larger than the document, but since most microcomputers have small screens, the ability of zooming the document to different sizes is needed.

It is important to remember that the basic rules of page design and aesthetics have not changed, just that new tools have been added. These new tools will help most people solve their publishing needs.

To choose a desktop publishing system that will fit a particular requirement; decide upon what type of publication will be printed, its audience, the budget, when and how the text and graphics are created, and how the document will be printed.

There are five major desktop publishing techniques:

1. Interactive Programs
   These programs will format text and graphics together on the computer screen by useing a menu driven command structure. The software can handle complex layouts and lends itself to division of labor by specialists. The big advantage of this type of program is the ability to see the layout before the document is printed.
2. Batch Formatting Programs
   These programs were the first computerized page layout programs. They work by entering individual codes into the computer. Batch formatting programs are good systems to work with to produce well defined and long documents. Many systems have tools for semiautomatic generation of the table of contents, index, footnotes and bibliography. Unfortunately the document cannot be seen until it is printed, and almost any change in the document requires that the entire document be formatted again.
3. Generic Coding
   These systems are generally less comprehensive than formatted codes. Due to the large amount of codes required, it is very difficult to create complicated formats, therefore a lot of manual cutting and pasting is needed to finish the document. Also with this system, the document can not be viewed until it is printed.
4. Keystroke Capture
   This system simply allows the raw text to be entered directly into a laser printer or typesetter. The finished document requires considerable manual paste-up.
5. Database Publishing
   These systems use an automated form of generic coding to produce a specific document with only specific changes. Since the basic format does not change, the format code is only entered once. The production of the document is completely automated. Any change in the format requires rewriting the program codes.

With the third generation of microcomputers now under development, future desktop publishing systems will offer superior software performance, screen resolution, processing speed, multiuser networking, and memory. These future systems will have capabilities of "real-time" on screen editing, full graphic capabilities for complex design tasks, and offer a combination of the above desktop publishing techniques.

Why Desktop Publishing

Traditional publishing methods involve a number of operations carried out by a specialist outside most companies or educational institutions; the graphic designer designs the visual look of the document, a typesetter sets the type, a camera operator creates halftone screens of photographs and artwork, a pasteup artist assembles the copy and art, and the printer prints the document. Now, desktop publishing can shorten this process, decrease the cost, and allow more control over the inhouse publishing process.

Inhouse publishing is not new; anyone with a duplicating machine or printer can publish. What is new, is how desktop publishing produces documents that look professional at a low cost.

In simple terms, desktop publishing uses the microcomputer to write, design and print documents.

Who Can Benefit

Almost anyone who works with the written word can profit from desktop publishing techniques. Everyone from corporations to individuals can publish their documents in-house. Since this decreases the dependence on outside typesetters and printers, it produces a faster turn around time. Documents are easy to update because there is no need to rewrite the entire document when there is a change of information. Below is a small list of the type of documents that are being created with desktop publishing.

1. Corporations
   Newsletters, reports, forms, and presentations
2. Universities
   Forms, applications, certificates,

curriculum vitae, syllabi, and course catalogs.

3. Businesses and Professions
   Letterhead stationary, invoices, business cards, resumes and correspondence.
4. Educators
   Class handouts, transparent overheads, grants, and papers.

In order to keep our students current with the expanding area of printing documents inhouse, a course was developed to introduce the basic techniques and advantages of desktop publishing. As the inhouse publishing field grows, so will the need for more trained people who can operate the computerized systems. Our design students will have need of computer operating procedures, a knowledge of digital typography, and mastery of graphic and publication software.

Information is increasing stored, handled, and transformed from paper into the electronic medium, which is changing the traditional materials of the graphic designer. This causes for some an uneasiness, because older and familiar design methods and materials are now transformed into electrical signals. Some designers that are comfortable with traditional page layout methods find it hard to accept the computer as a viable design tool.

Now that printers and typesetters are computerized, text and graphic alternations are easily accomplished. Combining innovative design capabilities with precise on screen editing of text and graphics, small publishing jobs can look professionally composed and printed. Gone are long periods of writing, editing, typesetting, cutting and pasting of mock-ups, and coordinating with the printer. Instead of paper, rubber cement, press type, composers, drafting table, the designer now works with digitized text and art, different screen formats, hardware, software, interfaces and the transferring of computerized information to the printer or typesetter.

Desktop Publishing and the Designer

Producing effective visual communication is the job of the graphic designer. Although desktop publishing allows the designer to design documents faster, it is his or her responsibility to produce a document that communicates effectively, and to communicate, a publication must have clarity, order, and meaning.

The basic design process does not change when working with the computer and publication software, (Definition of the Problem, Analysis, Alternatives, Solution, Production and Evaluation). The computer can be useful in the total design process. With publication software, the process of finding a design solution is impressively shortened. During the solution process, greater flexibility for generating ideas and their alternatives is permissible, changes can be done up to the last-minute before printing. Repetitive hand labor activities are computerized, making the design process more economical, allowing for a wider spectrum of ideas. A layout can be rapidly modified and reviewed, which means more ideas can be tried in a shorter amount of time than using traditional methodology, hopefully to produce better solutions.

In the beginning, there is the risk of allowing the technology to influence the output, but a good designer will use the computer as only a tool. With all its efficiency and creative advantages, desktop publishing itself does not guarantee quality publications.

The Classroom

I found out very quickly that teaching a new computerized medium to art and design students requires an extra amount of patience. It is both an awarding and trying experience.

Teaching the course requires a lot of research, new hardware and software products are constantly being released, just keeping current with this information is time comsuming, and reading product literature does not always give the complete picture. Also since this is a new design medium in our department, I have only myself to rely on for collecting pertinent information.

Through the first half of the semester, there is a dramatic difference between students. Some students learn the hardware and software operating procedures very quickly and try to push some of the software beyond their designed capabilities. Other students seem to have difficulty learning basic procedures and almost have to be "spoon fed" the information. Forgetting to practice good design principles the slower students have the tendency to blame their failures on the computer, forgetting that the computer will turn them into bad designers much faster then good ones. For the slower students, I require them to read the hardware and software manuals as they obtain their "hands on" experience. The students that become confident with the hardware and software are very helpful and they take pride in helping others. I find this rewarding, since I always tell the students that they can learn from each other and not to wait for me to tell them everything. I usually loose about two students a semester, either through poor study habits or due to my failure to excite them with the subject matter.

The Working Process

To control the subject matter and learning experience while working on a project, I remind the students of the proven procedures used to produce a traditional publication.

Publishing a document has common elements:

1. Start with a good idea and transform it into a printed document.
2. The essential tasks remain the same-designing the document, writing and editing the copy, producing the artwork, typesetting, layout and pasteup of the copy and artwork.

Sequence of Working

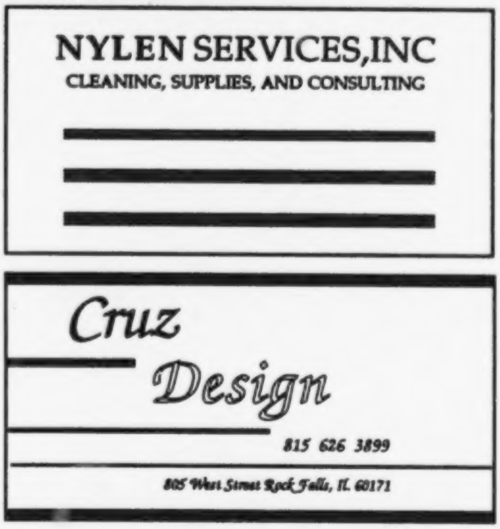
1. A well-thought-out visual design is the first and most important step. This will give a clear idea of what the document will look like, and it brings together consistent items such as logos, trademarks, column placement, headers, and footers.
2. Understand your audience. This will establish the guidelines for the editorial and art content. Knowing the audience will help to select a particular style in which to present the elements in a consistent manner through out the publication.
3. Know the reader's expectations.
4. Determine the physical size of the publication. Its page size, number of pages, and how many columns it will have.
5. Develop and finalize the copy and artwork before you begin the layout process.
6. Know how the publication will be printed.

The main concept of the course Desktop Publishing is to teach basic techniques in electronic publication, not to produce expert graphic designers. Because design principles are not covered extensively, prerequisite courses teaching the aesthetic principles of graphic design are required. The course is taught during a fifteen week semester with eighteen credit contact hours per week. Usually the number of students is limited to fifteen, because the lab has six computers. Because of the limited number of computers, all lectures and demonstrations are held with the entire class sharing the computers.

Lab time is broken up into time slots so personal help can be given to each student. The rest of the week the lab is open with student lab assistants in attendance. The hardware used is the Macintosh microcomputer, ImageWriter (dot matrix printer) for copy roughs, and LaserWriter (laserprinter) for producing the finished documents. The software used is MacPaint (Paint program), MacDraw (Drafting program), MacWrite (word processor), ThunderScan (digitizer for scanning art work), and PageMaker (publication software).

The first couple of weeks is mainly spent on learning the operating procedures of the computer. Due to the newness of the class, the projects are modified each semester after finding out what works and what dosen't. Five projects are completed during the semester.

The first project uses MacPaint and MacDraw to produce variations of a personal business card, trademark or logo. This project introduces the students to the graphic capabilities and font selections of the software. It also allows the students to mix text, line art and graphics together for the first time using a computer. Below in figure 1. are two student examples, both business cards were created in MacDraw software and printed out on the Apple Laserwriter. For some students, this project leads to them designing their own personal stationary for letters and resumes.



NYLEN SERVICES,INC
CLEANING, SUPPLIES, AND CONSULTING

Cruz
Design
815 626 3899
805 West Street Rock Falls, IL 60171

Figure 1. Tim Cruz, Northern Illinois University

In the next project the students learn basic word processing techniques by using MacWrite software. A cover letter is written to accompany a resume, some students add a personal logo or letterhead to the letter. During this project students are introduced to word processing skills. By the middle of the semester, I find them using the word processer for other courses and student organization announcements. When this happens I require them to supply their own paper and printer supplies. I found that if I do not keep a tight reign on the students, they will deplete the printer paper and ribbons in a very short time. Also when they are required to print their projects on good quality paper (which they have to supply), the students are not in such a hurry to print their image every time they make a change to their project. Using their own paper influences more proof reading.

In the third project, the students design a resume using PageMaker. Most of the students in this class have never designed a resume, so it serves as a good project to introduce publication software. The project makes them think about writing their own copy, editing (proof reading), and developing variations to a page design. The image in figure 2. is a student resume, the project was completed using MacWrite, MacDraw and PageMaker software. I found that most students will use this project to design the resume that they will use when they graduate. Some students also return within a year after they graduate to ask for permission to use the lab to revise their resume.

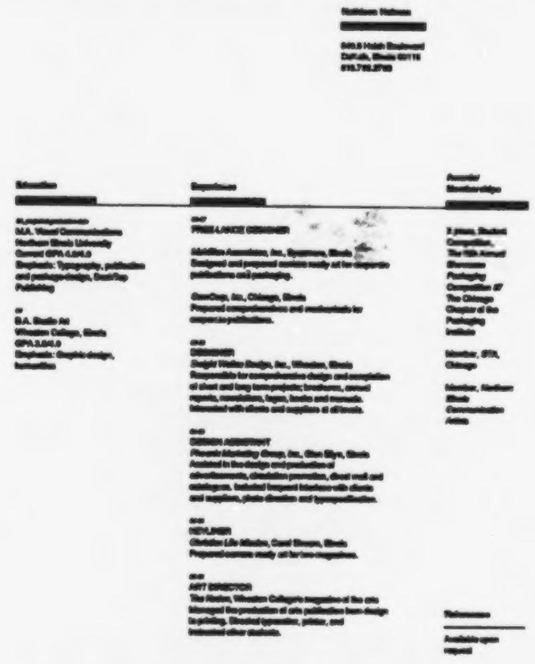
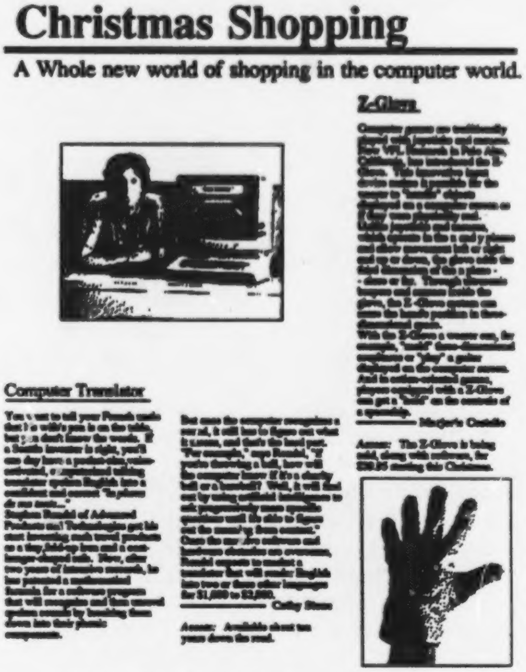


Figure 2. Kathleen Holman, Northern Illinois University

Next a newsletter is produced, using all of the above software to combine text and graphics into a finished document. By the end of this project the students should be familiar with most of the techniques of PageMaker. Figure 3. shows one sheet of a newsletter project. The copy was written in MacWrite, the images were scanned off photographs using the ThunderScan digitizer, the line art and layout was completed with PageMaker software.



Christmas Shopping

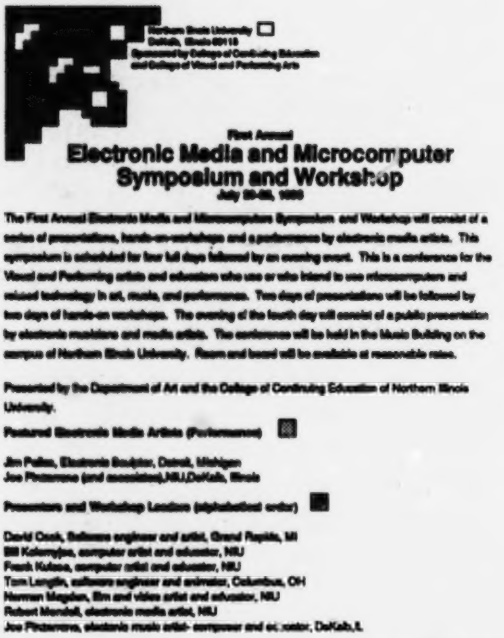
A Whole new world of shopping in the computer world.

Z-Glove

Computer Translator

Figure 3. Sue Basak, Northern Illinois University

Finally the semester ends with the students producing a document of their choice. Allowing them to pick their own project has produced mixed results. The good students use this opportunity to experiment with the software, others "kill two birds with one stone" by using their new computer skills to complete a project from another graphic design class, and some who have freelance jobs develop a design solution for their customer. Figure 4. is the first page of a five page brochure designed for this project. The copy was created in MacWrite, most of the graphics were created in MacDraw, and the brochure



First Annual

Electronic Media and Microcomputer Symposium and Workshop

Figure 4. Kari McGlinnen, Northern Illinois University

composed in PageMaker with line art added.

All final outputs are printed on the LaserWriter which has a resolution of 300 dots per inch. If a student wants letter quality output, the document can be sent to a Linotronic typesetter. Also if a student wants to drop-in colored art or high resolution halftones, that is allowed.

When each project is due, a group critique is given. This gives the students the opportunity to talk about aesthetic issues and their experiences with the equipment, discussing software and hardware abilities and limitations.

After the students become comfortable and familiar with the computer, they discover that experimentation is quick and easy. Since the effects of their changes are immediate, they can see what the document looks like as it develops. Having this new ability to quickly change text and graphics with ease will sometimes change the way they use text. Unfortunity having this freedom will cause them to forget basic typography and graphic design skills. The most common error that occurs is that the document will have too many different type of font faces, sizes, and styles.

Sometimes the experimentation is carried to the point where the software will crash. Although this puts the students off, they soon learn that the computer is just a machine, and that they have to work within its limits. Most students enjoy taking advantage of the computer's speed to modify and change their design ideas. Some students play it safe to the end of the semester, and never accept the computer as a new and exciting design tool.

Due to its lower cost, friendly user interface, software support, and graphic capabilities, I choose to teach the class using the Apple Macintosh microcomputer and Apple LaserWriter as the primary hardware devices. Figure 5. is a diagram of the hardware system used to create and print student work. The computer is a 512K Macintosh computer with two single sided disk drives. The dot matrix printer is Apple ImageWriter and the Laser printer is Apple LaserWriter (300dpi). If numerous documents are printed, a Macintosh computer with a 10meg internal hard disk is used.

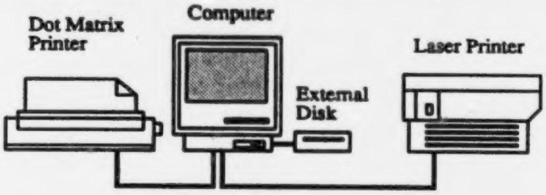


Figure 5.

Basic Software Needed

Desktop publishing software have very limited text processing capabilities, therefore a word processor is needed to write and set text. For producing art, "draw" and "paint" programs are required for ample graphic tools. If business charts and graphs are generated in large numbers with constant revisions to data, then a business graphic package would be valuable. To capture three or two dimensional images, a digitizer or scanner with graphic tools would be needed. Most inexpensive digitizers and scanners do not produce the halftone quality of photographs. If photographs are used, locations are reserved in the document, later the photographs can be placed into these locations before sending the camera ready document to a professional print shop.

If large or numerous small documents are produced, disk memory is quickly filled. To eliminate the nuisance of removing and replacing file disks, a hard disk is needed. Also the new double sided internal and external disk drives are helpful to eliminate "disk shuffling".

Figure 6. shows the basic steps taken for most desktop publishing systems. The text and graphic

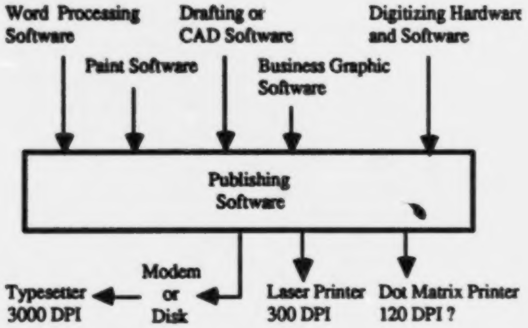
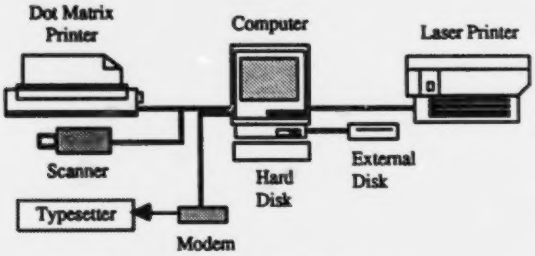

Word Processing Software
Drafting or CAD Software
Digitizing Hardware and Software
Paint Software
Business Graphic Software
Publishing Software
Modem or Disk
Typesetter 3000 DPI
Laser Printer 300 DPI
Dot Matrix Printer 120 DPI ?

Figure 6.

files are generated in their related programs, and then the files are combined into the publication software. When the document is complete, the information is sent to a variety of printers for hardcopy output. The software in figure 6. can be added or subtracted to, depending upon what type of documents will be printed, expectations of the audience, and the needs of the system user..


Dot Matrix Printer
Computer
Laser Printer
Scanner
Hard Disk
External Disk
Typesetter
Modem

Hard Disk (10 Megabytes)
Publishing Software
Word Processing Software
Graphic Software
 1. Paint Software
 2. Drafting Software
 3. Business Software
 4. Clip Art Software
Scanner Software
Picture Library Files

Figure 7.

Figure 7. is an example of a maximum integrated desktop publishing system for a microcomputer. The Hardware and software shown is not needed in all desktop publishing systems, nor do all the peripheral devices need to be connected at the same time. Figure 7. shows a system designed after speaking to former students and designers using desktop publishing systems. As new hardware and software enter the market, the above system will be modified as the publishing needs change with the audience.

Computer Graphics

Most publishing efforts can probably use a number of graphics programs, but before a graphics package is purchased, decide if it is needed. If the purpose of the publication is to give information and to try to persuade the audience, it is useful to use both the written word and visual images.

For most general uses, the software package would contain compatible draw, paint and business graphics programs. Compatibility is very important, because a graphic image will often be created using features from different programs. Most graphics programs help greatly with the mechanical details of creating illustrations, and make it easy to edit, reuse, and manipulate images.

Until better graphics programs are available, most generated computer graphic images fall below what most designers accept as quality images. A good graphics program and laser printer will produce a image with a 300 x 300 dot per inch resolution. This is acceptable to some and unacceptable to others, depending upon their own and their audience's expectations.

Most computer graphics programs fall within four major types:

1. Business Graphics Software
    - Pie Charts
    - Bar Charts
    - Line Charts
    - Maps
2. Paint Programs
    - Bit-mapped images
    - Free form drawing
    - Brush and air brush techniques
    - Control of each picture element
3. Draw Programs
    - Object oriented images
    - Geometric shapes
    - Sharp lines and arcs
    - Scaled and structured drawings
4. Computer Aided Design Programs
    - Extremely detailed drawings
    - Automatic dimensioning
    - Scaled drawings
    - Architectural and technical drawings

Word Processing Software

In the future, publication software will most likely offer complete word processing capabilities with very good graphic tools to create and modify graphic images. But for now, a word processor is needed to produce the copy for a document, therefore certain features should be available.

Basic Word Processing Features

1. Multiple Line Headers and Footers
   The top and bottom margins of a document which repeats on every page. It can include text, pictures, and page numbers.
2. Automatic Pagination
   Automatic insertion of page numbers on every page of the document.
3. Search and Replace
   The ability to search and edit text.
4. Cut and Paste
   Selected text, pictures, rulers or page breaks can be removed and replaced anywhere through the document.
5. Word and Character Count
6. Table of Contents
   The automatic generation of a table of contents.
7. Option of Various Fonts and Sizes
   The ability to choose different fonts, type sizes and styles within the document.
8. Ability to Incorporate Graphics
   Artwork created with compatible graphic software can be transferred and placed within the document. This ability is very important for interactive systems.
9. Kerning
   The ability to control spacing in points between letters and words allows for professional looking documents. A point is the printer's smallest unit of measurement. There are 12 points to a pica, and there are approximately six picas to the inch.
10. Hyphenation
    Dictionary-based hyphenation is useful for producing even columns.

Spelling Checker Software

If your word processor does not correct spelling, a spelling checker is an important addition. Checkers can work as accessories or as a stand-alone addition. They can review a whole document or a block of text, others are interactive, looking for errors as you type. The size of its dictionary will influence its speed and memory capabilities. Be aware that the size of the dictionary can be misleading. Some manufactures count root or variations of root words, omit common words, flag contractions or plurals as errors. Others will allow new words to be added to its dictionary. Special features are available; thesaurus, glossary, proper hyphenation, but the most important feature is how fast it catches misspelled words and how few correct words it flags as a misspelling.

Typeset Font Software

A typeset document is more impressive and has greater authority and easier to read then a typewritten document. The typeset document will command attention because it implies that time and care went into its creation. Also typeset text is compact, so more information is placed on the page.

If the supplied software and printer fonts are not enough to handle design needs, downloadable fonts (temporarily loaded) can be added to the system. Building a large font library can be a serious investment, but by mixing font sizes and styles of just a few fonts, many text variations can be created. In choosing a font, one should look at its aesthetic quality, ease of use and especially its design versatility.

Some Basic Font Faces

1. Text faces (serif).
   With these faces the eye is able to scan the document quickly, also the typefaces have tapering strokes that are terminated with serifs that allow for easy recognition of all letterforms. Text faces are used for lengthy passages of running text.

Times Roman	**Times Roman**
Palatino	**Palatino**
Bookman Plain	**Bookman**

2. Display faces (sans serif)
   These typefaces have almost uniform letter thickness which can create readability problems over a large area of text. *Sans serif* faces with light strokes can cause printing errors in laser printers when using some point sizes. Some pixels are removed during printing, changing the appearance of the text. Display faces are used in short passages such as titles, letter heads, cover design. The eye is attracted by their bold and dramatic style.

Readability is taken care of by the type's page placement and its larger face size special function when the type itself is needed as part of the narrative. Decorative faces are too distinctive to blend well with informational typefaces. These typefaces do not fit any particular catagory. They may be in outline form, irregular in shape, or they may be another redesigned typeface with extra swashes and embellishments added to the characters.

Venice Venice

Venice Shadow

It is important to choose fonts that can perform different functions. A display font can sometimes serve as a text font and a text as a display font. Choose a font style that comes with a large number of point sizes. Remember that a font's stroke weight should be consistent, the alignment of characters should be precise and all letter spacing should be even. Examine the size ranges, what looks good in one size, may not in another.

Printing time between typefaces vary, especially if you have downloadable styles. Also some downloadable fonts take up a lot of memory which slows down printing time or can cause crashing if memory areas are overlapped. Future printers with larger memory areas will solve this problem.

Copy protection can sometimes be a problem if the fonts are only imprinted to a particular printer. Also it may cost more if an on-sight license is needed for multi-users of a system.

Publishing Software Buying Checklist

Certain basic items should be looked for when purchasing a good publishing program. The designer working with the software should be comfortable with its operating procedures. Try the software out before purchasing, do not rely on someone elses' opinion. Remember, there is no such thing as the perfect software package, finding one that fits your needs is difficult at best. Each software package has its own particular strengths and weaknesses, and each finds its own supporters and opponents.

Basic Page Layout Features

1. Text Processing
   The software should allow for fast opening of text documents while in the page layout program. What options (deleting, adding, changing text, setting tabs, moving one word or whole blocks of text) are there to format text? How does the program accommodate multipage edit changes, does a change on one page effect changes on another? Can text be wrapped around a variety of shaped graphics? New publishing software will most likely contain better word processing capabilities.
2. Graphic Processing
   The software should allow for fast opening of graphic documents while in the page layout program. How are graphics handled for cropping and resizing images? What type of drawing tools (different size paint brushes, erasing, zooming, etc.) and line variations (size and texture) are available? Do all the graphic tools work in all display sizes? Is there the ability for adding special graphic effects?
3. Page Processing
   Can master pages and templates be created? How easy is it to insert, delete and swap pages? It is helpful to create and print several thumbnails on one printout. Do changes on one page automatically cause changes to the rest of the document? Must you save the whole document, or can you select certain pages to be saved?
4. Performance
   What is the printing speed when text and graphics are mixed? How fast can documents be opened? How does your working style agree with the program's interactive operating design? Is the working style similar to traditional page layout methods? Is output page size limited or is there the ability to choose a variety of printing sizes?
5. Display
   The screen in actual or expanded zoom display should show exactly what will be printed. Are there different zooming capabilities for accurate placement of elements? Can you do all the needed editing, placing, sizing and cropping on the screen? Are there guides and snap-to lines for controlling element placement? Can ruler measurements be set to points and picas allowing for photographs to be dropped into the printed page? Some displays will show ruler measurements in

inches, centimeters, points and picas.

6. Hyphenation and Justification
   Some programs have manual and automatic hyphenation. Justify, aline center, aline right and aline left is important.
7. Leading
   Some programs have the ability to adjust line spacing in points between any line or lines of type, allowing for more accurate spacing within columns.
8. Kerning
   The ability to control spacing in points between letters and words allows for professional looking documents. A point is the printer's smallest unit of measurement. There are 12 points to a pica, and there are approximately six picas to the inch.
9. Support
   What are the warranty terms? How much time is allowed to learn the software? What are the copy right laws and are the disks copy protected? Is the supporting documentation easy or hard to understand?
10. Transferring Information
    The system should operate with an adaptable communication language. This allows documents to be transferred to different printers and typesetters from disk or across modem.
11. Length of Document
    Some software only allow one page to be produced, others will generate enough pages to publish small magazines.
12. Copyright Protection
    Different software products have a variety of restrictions. Some allow their disks to be copied, others do not. Usually the more expensive software will need a master disk to start the program. On-sight license may be needed for multi-user systems.

In the future; hyphenation, justification, leading and kerning will be available on most quality page layout software. Also publication software will most likely offer complete word processing capabilities with very good graphic tools to create and modify graphic images.

References

Books

The Design of Advertising, 5th ed., Nelson, Roy Paul, Dubuque, Iowa: Wm. C. Brown Publishers, 1985.

Desktop Publishing on the Mac, Kevin Rardin, A Plume/Waite Book, New York, New York, 1986.

The Complete Desktop Publisher, Daniel J, Makuta and William F. Lawrence, Compute! Publications, Inc., Greensboro, North Carolina, 1986.

The Graphic Designer's Handbook, Alastair Campbell, Running Press, Philadelphia, Pennsylvania, 1983

Magazines

Publish!, 555 De Haro Street, San Francisco, CA 94107. Published bimonthly.

MacWorld, 555 De Haro Street, San Francisco, CA 94107. Published bimonthly.

MacUser, 25 West 39th Street, New York, NY 10018. Published monthly.

Articles

Whitney, Patrick. "Information, Computers and Design." Design in the Information Environment. 1st ed. New York: Alfred A. Knopf, 1985.

Thompson, Keith. "Pagemaker Remains Chief Page Composer." InfoWorld, June 9, 1986, pp. 39-40.

Ebbesen, Ebbe B. "Desktop Publishing." MACazine. April, 1986, pp. 16-27.

Cavuoto, James. "Sophisticated Page Makeup Gets Personal." Computer Graphics World. September, 1985, pp. 49-52.

Burns, Diane, Venit, S. "Revised Editions." Publish. (Macworld. July, 1986, pp. (28-33.)

PERSPECTIVES ON LOGO TEACHER TRAINING

Doug Clements, Chair
Kent State University
Kent, OH

ABSTRACT

Teacher training is a critical concern in the development, growth, and use of Logo in the schools. This concern is composed of many issues, both general and specific, including attitudes toward computers, time available for study, and expectations of Logo.

Frequently, university programs of study do not provide a separate Logo course. Instead, Logo is mentioned in passing in one or more courses, or perhaps one or two evening Logo workshops are presented. This leads to a variety of serious misconceptions about Logo, such as: (1) Logo is drawing pictures, (2) Logo is just a programming language, (3) Logo is not relevant to the day-to-day business of classroom teaching, (4) Logo is easy, and (5) Logo is difficult.

Panel members will comment on these issues from their individual perspectives and will offer suggestions for effective Logo teacher training programs, including how to show teachers that:

1. They can develop the ability to use Logo's power effectively, but it will take time and effort;
2. Although there is an important place for teaching philosophy in using Logo, the ideas in *Mindstorms* should be discussed and subjected to criticism and contemplation;
3. Expectations for immediate outcomes may be too high while expectations for the amount of time, resources, and support required for these outcomes are too low;
4. There are hundreds of ways Logo can be used in the classroom; Logo is not necessarily an end in itself, but can support learning in many other disciplines;
5. Attention to good Logo programming style can benefit students in many different areas;
6. Teachers need more metaphors to better understand the Logo langauge;
7. A grounding in problem solving processes is important so that teachers will know how to support their students when they encounter difficulties and are frustrated;
8. Students should not necessarily be required to create all of their required procedures; teachers should provide students with appropriate tool procedures, sample problems, and challenges within a rich learning environment;
9. Although teachers can learn Logo alongside their students, they need to know at least as much about Logo as their students and at a deeper level to be most effective;
10. Learning about Logo can give teachers insight into their learning styles and those of their students;
11. Follow-up on Logo skills is important;
12. Logo cannot work in the schools in isolation; widespread change in schools and universities is necessary for its full actualization;
13. It is important to inform superintendents, supervisors, and other decision makers about the value of Logo in the curriculum; and
14. Learning about Logo is exciting, frightening, sometimes disappointing, not guaranteed, but ultimately extremely satisfying.

PANELISTS

Judith Harris
University of Virginia
Charlottesville, VA

Dan Watt
Educational Development Center
Newton, MA

Molly Watt
Educational Development Center
Newton, MA

SPONSOR: ICCE and Logo Exchange

MATERIALS DESIGN

Ronald Owston, Herbert Wideman, Curt Dudley-Marling
Nancy Houston
Debra Stein, John Knauer

ABSTRACT: Software Evaluation: How Do Teacher Reviews of Software Compare to Classroom Results?

Ronald D. Owston, Herbert H. Wideman, and Curt Dudley-Marling, York University, Toronto, Canada

To date, most software evaluations have been undertaken by teachers using some form of written checklists or review procedures. While some evaluation models encourage reviewers to observe student use of the software being assessed, this step is not often implemented in a systematic way. Little is known about the ability of teachers to prejudge the utility of educational software or the range of students' responses to it.

The present study has been designed to address this question by comparing teacher and field evaluation data on more than 100 commonly used K-12 software packages. Panels of teachers are evaluating the software using the criterion-based Educational Software Evaluation Scales to reach a consensus on its quality in four key areas: pedagogical content, instructional presentation, documentation, and technical characteristics. The panels are also writing brief narratives on the strengths, weaknesses, and unique features of the software. The classroom trials are being conducted by a different set of teachers who field test the packages using prescribed guidelines.

The panel evaluation and field-test data are presently being analyzed using the constant comparative method. The goal of the analysis is: (1) to determine the extent to which the panel evaluations are congruent with the field-testing results; and (2) to identify factors such as level of student interest and efficacy of instructional design in which the two sets of data diverge from each other.

Preliminary results indicate that while teacher assessments anticipate many of the field-test findings, classroom trials do offer useful new information about technical and design limitations, program operation, and the effect of different contexts on outcome.

ABSTRACT: Using Interactive Videodisc in an Introductory Computer Class

Nancy P. Houston, Chairman, Computer Systems Department, Grove City College, Grove City, Pennsylvania 16127

What better way to teach about the computer than to allow the computer to teach you itself? This is the approach used at Grove City College in a course entitled Computer Introduction and Applications.

This presentation will describe the use of the DEC IVIS system (Level III interactive videodisc) as a teaching method. The course has been taught for the past six semesters using the videodisc as the instructional method for delivering the equivalent of one month of standard lecture instruction.

Advantages and disadvantages of using a videodisc as the primary means of instruction will be discussed. Results of an attitudinal survey of student satisfaction with the delivery method as well as research results comparing lecture/disc and lecture/tape instruction will be presented.

ABSTRACT: CAI Programming and Student Learning Styles: Matches and Mismatches

Debra K. Stein, Ph.D. and John Knauer, Pennsylvania State University, The Behrend College

Although CAI has developed in an effort to individualize the educational environment for the student by supplying learning situations that are visually stimulating, self-paced, and immediately reinforcing, the application of CAI has occurred without consideration for the students' individual learning styles. As Dunn and Dunn (1978) have shown, not all students rely upon the same perceptual modality when they learn, nor do they learn best in the same types of social situations. Thus, the blanket use of one teaching technique or classroom set-up should never be considered to be the optimum technique or set-up for all or even the majority of students.

The project currently underway at Penn State Behrend is one that is attempting to assess if a match between student learning styles and CAI curricula will result in greater achievement gains than a mismatch. Approximately 90 middle school students have been administered Dunn and Dunn's Learning Style Inventory and divided into groups identified as auditory, visual or tactile learners. These groups are being administered CAI programs in science that either match or mismatch their preferred learning modality. Pre- and posttest science achievement scores will indicate if students learning with programs that match their preferred modality will show greater gains than those students working with programs that don't match their preferred modality.

UNIVAC, THE EARLY YEARS

Joanne W. Iverson
Iverson Associates, Inc.
Philadelphia, PA

Stephen Longo
LaSalle University
Philadelphia, PA

ABSTRACT

These UNIVAC pioneers will gather to reminisce about their early endeavors to cope with the emerging technology, and to talk about the feats of genius they witnessed.

The first computers were born of the need to solve problems in ballistics. The ENIAC (Electronic Numerical Integrator and Computer) was the first announced in 1946, followed by the EDVAC (Electronic Discrete Variable Computer). The BINAC (Binary Automatic Computer) was completed in 1949 and was small enough to be operated inside an airplane in flight. The UNIVAC (Universal Automatic Computer) was completed in 1951 and marketed soon after by Remington Rand. The folks who will gather for this historic occasion are those who worked for and with J. Presper Eckert and John Mauchley on these history-making projects.

PANELISTS

Raymond Frankson
University of Pennsylvania

Howard Masters

Rick Cloughly
Sperry

Paul Chinitz

Cedrick Lee

Saul Brownstein

Betty Jean Bartik

Jerry Smoliar

Al Tonik
UNIVAC Retirees Club

Rich Berryman

Jerry Feldscher

Laura Cousins

Jim McGarvey

Fred Frankel

COMPUTER-ASSISTED LANGUAGE INSTRUCTION

Randall Jones, *Chair*
Brigham Young University
Provo, UT

ABSTRACT

(1) *From Drill and Practice to Contextual Input in Second Language CAI*
At Duke University we have recently been adapting our existing German CAI "drill and practice" format to a "contextual" delivery in order to be used in more intensive language programs. A typical contextual drill might consist of: 1) A "scanning phase" in which questions are asked about a text on the screen. The questions are sequential and the answers are marked to facilitate acquisition. 2) A "sequencing phase" in which the sentences of some paragraphs are scrambled and the student has to find the correct order. 3) A "close phase" in which grammatical or lexical items are removed to be filled in random order by the student. 4) A "chaining phase" in which the text is presented with certain connected elements highlighted. 5) A "matching phase" in which a list of words is matched with words marked in the text. The theoretical basis is to emphasize and facilitate "acquisition" of whole texts rather than to drill and practice vocabulary or drill grammatical concepts.

(2) *Computer-Assisted Language Learning: Is It Succeeding?*
Is computer-assisted language learning (CALL) having the impact on everyday teaching and learning of foreign languages that many thought it would some years ago? This presentation will focus on both sides of the coin: the obstacles that hinder the progress of CALL, as well as the reasons for believing that many obstacles are being, or will be, overcome. We shall consider both sides of the following topics: courseware, hardware, implementation, financing, the contribution of commercial enterprises, the quantity and quality of teacher involvement as well as the professional rewards for such activities.

(3) *Computer-Assisted Language Instruction with Sight and Sound*
The basic microcomputer is limited in its application to the learning of languages. With a keyboard for input and a screen for output the types of activities are limited to the manipulation of written material. However, additional hardware makes it possible to incorporate both audio and video into a CAI language program. Audio allows the use of listening comprehension lessons and tests as well as pronunciation practice. Video, either still or motion, provides a realistic context for a lesson. Because the audio and video material can be accessed randomly by the computer, the same degree of versatility and flexibility is possible as is the case for standard CAI programs.

PANELISTS

Peter Batke (1)
Duke University
Durham, NC

Glyn Holmes (2)
University of Western Ontario
London, Ontario

Randall L. Jones (3)
Brigham Young University
Provo, UT

SPONSOR: ACH

255

RESEARCH ABOUT EFFECTIVENESS: SELF-DECEPTION AND REALITY IN SURVEYS, PROGRAM EVALUATION, AND FIELD EXPERIMENTS

Henry Jay Becker
Center for Social Organization of Schools
Johns Hopkins University
Baltimore, MD

ABSTRACT

How effective are instruction programs that use computers? To answer this question, we have many modes of inquiry: We react to claims made in commercial advertisements and product reviews. We try out programs and intuit their value. We develop evaluation checklists and apply them to particular programs and circumstances.

More formally, we conduct research that attempts to go beyond the admitted bias of commercial producers and their distributors and beyond the limited perceptions of individual computer using teachers. For example, we conduct national surveys of computer using teachers and report their perceptions of how their own use of computers in their classes has affected their students' performance. Or, as designers of a program of instruction using computers, we measure students' performance under the program and perhaps compare that performance with other students or with their prior performance. Or, we might design full-blown experiments in which students, and perhaps teachers, are randomly assigned to alternative computer and noncomputer methods for accomplishing a certain domain of learning and compare the results of using these contrasting approaches.

Each of these methods has its advantages and disadvantages. This presentation will compare what we can learn from different ways of approaching the question: "How effective are instructional programs that use computers?" The basis of this discussion will be: (l) findings from the 1985 National Survey of Instructional Uses of School Computers, a survey conducted by the presenter among more that 2,000 schools and 6,000 computer using teachers; (2) an analysis of recent professional literature on the outcomes of using computers for instruction in K-12 subject areas, including dissertations, published articles, and school system program evaluations; and (3) a design for a series of systematic field experiments, one of which is now underway, that will greatly magnify our cumulative knowledge about the effectiveness of instructional programs using computers.

The comparison of alternative methods for learning about the effectiveness of computer based instruction will focus on problems endemic to distinguishing between actual effects of instructional practices in educational settings and "artifacts"-- apparent consequences that in fact have other causes.

For example, in observing what teachers and others say about their use of computers, one finds that often teachers talk about how enthusiastic students are, but they are less likely to report actual differences in learning that they perceive to be a result of using computers. Or they identify certain children who have clearly benefited from using computers but don't address the question of whether, on balance, across all class members, the students are better off using their new instructional program than they were before. Or, although most children are learning using computers, perhaps the computer using teachers are generally good instructors who would have succeeded just as well had they used other methods; or the students are exceptionally bright learners; or the conditions under which the computers were used were unusually resource intensive. Or, by using computers, perhaps certain curriculum elements are emphasized at the expense of others that are necessarily downplayed because of time constraints or because those aspects of instruction are not emphasized by the software. Evaluation methods are necessary that take all these factors into account.

The goal of research is not to depreciate the value of the tremendous investments that people are making into improving students' intellectual developments and academic performance by means of computers. No, the goal of research is to understand, without bias, the actual impact of instruction using computers and how that impact varies under different circumstances, for different students, using different materials, so that efforts to use computers to improve student learning can be channeled in the most productive directions possible.

TRAINING EXECUTIVES WORLDWIDE

Sandra Morris, *Chair*
David Sarnoff Research Center, Inc.
Princeton, NJ

ABSTRACT

Rapid changes in the capability and uses of technology in the workplace have introduced new areas of interest in the world of industry and training. On the one hand is the challenge of keeping adults in the workplace up to date on new advances and new technologies; on the other are the creative solutions these advances and technologies offer to the basic business of training and teaching professionals.

This session draws on the experiences of three professionals from industry who are charged with training executives and managers in business. Each panelist has had distinct experiences in the area of technology, and in particular, computer and videodisc use in training. An emphasis will be placed on the unique problems and solutions posed by training managers in a variety of settings. The panelists have had experience in planning life-long education programs for business graduates; planning and implementing short but intense residential training programs for executives; setting up computer and management training centers in remote international settings such as Egypt, Jamaica, and Indonesia; the design and use of videodiscs in training for data processing, management, sales, and manufacturing. Curriculum lessons garnered from years of experience will be shared and discussed. New problems and questions will be posed.

PANELISTS

Laverne Cook
RCA Government Services
Cherry Hill, NJ

Tom Kuhn
University of Pennsylvania
Philadelphia, PA

Harry Lasker
Interactive Training Systems
Bedford, MA

257

SPECIAL NEEDS

Jo Lynn Autry Digranes
Phil Mackall
Leon Foster
Antonia Stone

ABSTRACT: Development of an Individualized Computer Curriculum for Gifted/Talented American Indian Students

Jo Lynn Autry Digranes, University of Oklahoma, Norman, Oklahoma

This curriculum project was the computer education component of a series of weekend sessions and a four-week summer session for American Indian high school students in Oklahoma and Kansas. The sessions were sponsored by American Indian Research and Development, Inc., of Norman, Oklahoma. Financing came from public and private grant funds. The sessions included a variety of subject areas: creative expression, drama, art, mathematics, science, outdoor education, leadership, dance, gymnastics, music, Indian philosophy, media production, and computer education. Emphasis was on identifying gifted/talented American Indian students and on providing motivation and enrichment activities for those students.

The computer education component, developed by the presenter, focused on the development of individualized education plans (IEPs) for each of the computer students. Plans were designed based upon student abilities and interests and upon available resources. Students were encouraged to select projects of interest to pursue during the sessions. Objectives were written for each pupil with evaluation performed through written tests, teacher observation, student self-reports, and judgement of student products. Computer languages, computer applications (word processing, data bases, spreadsheets and graphics), and computer assisted instructional software were included in the curriculum, which was designed to introduce new concepts and to strengthen existing skills.

To enhance retention and transfer, the computer education activities (and the other course offerings) were linked to other subject areas and to American Indian tribal applications such as tribal government uses for business software.

ABSTRACT: CMES and the Hearing Impaired, New Ways of Sharing Computer Technology

Phil Mackall, CMES Project Leader, Pre-College Programs, Gallaudet University, Washington, DC 20002

Five years ago, Pre-College Programs began planning ways to use computers to meet its own needs and the needs of other programs with hearing impaired students. In addition to the software used for data management, word processing, and CAI functions, three distinct products/services have evolved that will benefit nationwide educators in this field and may also help educators with slightly different populations. These products are: (1) Software Evaluation Clearinghouse and Lending Library for Educators of the Hearing Impaired; (2) Software Shopper; and (3) KendallNET.

The Software Evaluation Clearinghouse and Lending Library evaluates software for use with hearing impaired populations and lends software to qualified programs to evaluate or to use in the classroom. There are approximately 275 titles available now. The Software Shopper is a catalog of approximately 350 selected public domain software programs for Apple computers. The software can be loaded onto a disk in easy-to-use menu fashion for a minimal charge. KendallNET is an electronic communication link between schools and programs serving the deaf, which accepts communication from any standard computer with modem. Both personal mail and bulletin boards are provided. Administrators, teachers, and other professionals can share professional development opportunities, teaching ideas, software and hardware information. Students can send paperless pen-pal notes around the nation in an instant.

This presentation will discuss the current status of each of the products, including function, benefits, and accessibility to the general public. Handouts regarding each will be provided.

ABSTRACT: Outfitting the Expedition: Equipping the Special Education Student with Lifetime Skills

Dr. Leon Foster, Computer Coordinator, Bay Trail School, Penfield Central Schools, 1760 Scribner Road, Penfield, New York 14625.

This presentation will deal with building a matrix of lifetime skills for middle school special education students in grades seven and eight. Using *LogoWriter* and simulation software, students develop problem solving skills, word processing and data base management skills.

In a cooperative program between the special education and computer teachers, special education students use computers as tools in the context of daily experience for enhancing the development of cognitive skills and strategies.

Students use LogoWriter to keep a written and illustrated journal of experiences, thoughts and feelings. Student work is merged with teacher programming to form an interactive class project. In the LogoWriter environment these students combine written work with graphics; convey their ideas, thoughts and feelings to others; and build confidence in handling fundamental survival skills--in the present and for the lifetime expedition to come.

ABSTRACT: Computers in Correctional Education: Report of a 1986 National Survey and Recommendations

Antonia Stone, Executive Director, Playing to Win, Inc.,
106 East 85th Street, New York, New York 10029

If increasing computer use in corrections is to be a significant approach to closing the gap between the educational deficiencies and needs of inmates and the correctional education programs currently available, then first, a thorough investigation of the correctional experience with computers to date needs to be made. Second, based on the results of this investigation and on the broad use of computers in education generally, guidelines must be developed that could serve as a resource for correctional educators wrestling with the problems implicit in the implementation of computer learning technology. These were the assumptions on which the project completed by the Correctional Education Association and Playing To Win was predicated.

With funding from the National Institute of Corrections, the Correctional Education Association and Playing to Win jointly undertook a survey of computer use in correctional education. Our first effort was to identify every state and federal adult facility in which computer use had been implemented. After identifying these 152 institutions, we sent surveys to their instructors and administrators. From the surveys returned (82 percent), we selected five sites for visits by project personnel.

This presentation will report on the survey and the site visits, and share the resulting recommendations of the project staff.

VIDEODISC INSTRUCTION IN MATH AND SCIENCE

Douglas Carnine
University of Oregon
Eugene, OR

ABSTRACT

The session will present the rationale and procedures for videodisc courses in mathematics and science. The courses reflect the workings of a team consisting of an instructional designer, content expert, videodisc expert, and television production team. The session will illustrate the contributions of each team member: (1) research on the effects of various instructional design principles for reducing errors, increasing transfer, etc.; (2) demonstrations of how random access, picture stops, still frames, etc., are used; (3) dynamic video segments illustrating various scientific principles and phenomena; and (4) the way in which a teacher uses a remote control to present a course to an entire class. Segments will be drawn from chemistry, earth science, fractions, and advanced word problems courses.

UNIVERSITY COMPUTER SCIENCE

Victoria Dixon
Elsa V. Parsegian, Howard W. Pullman
Richard A. Alo'

ABSTRACT: An Investigation of Prior Sources of Difficulties in Learning University Computer Science

Victoria Dixon, Computer Science Department, Canisius College, 2001 Main St., Buffalo, New York 14208

The primary purpose of this study was to investigate various sources to determine their relationship to the achievement of beginning university computer science students, particularly those sources that contribute to student understanding of what a computer is and does and those that affect student ability to handle the basic symbol manipulation necessary to process information correctly.

A criterion test (dependent variable) designed by the investigator was used to measure some basic concepts and some frequently made programming errors. A total of 197 students from five different classes at the University of Buffalo participated in this study. It was shown that there was no significant difference for students who used BASIC, FORTRAN or Pascal as their initial programming language. It was also shown that students who began computer work in grade school and those who wrote longer program code did have some advantage over those who did not. Other independent variables used as possible contributing factors for achievement were previous mathematics achievement and learning style preferences. The program analysis did suggest that there is greater emphasis on program structure, rather than on symbol manipulation, in some computer science classes.

ABSTRACT: Sex Differences in Attitudes Toward Computers

Elsa V. Parsegian and Howard W. Pullman, Youngstown State University, Youngstown, Ohio

This presentation reports selected results of an ongoing study of correlates of students' attitudes toward computers and their place in society as well as correlates of success in an introductory computer information systems course. Students complete a survey that assesses their attitudes toward computers--via a 35-item Semantic Differential Scale--and their attitudes toward the place of computers in society--via a 10-item Likert scale. Students also supply information about past computer experience and the type of such experience. Approximately 400 students have completed this survey in the past year. The results reported here reflect the study of differences in attitude on the part of male and female students.

The literature abounds with reports of sex differences in attitude toward mathematics, computers, and quantitative/technological fields in general. Preliminary results in our study seriously challenge previous findings of pervasive negative attitudes of females toward such fields. Indeed, those differences that we find to be statistically significant show female students to be more positive toward computers than male students. Females characterize computers as more pleasurable, humorous, sensitive, spacious and public. Males characterize computers as more painful, serious, insensitive, constricted and private. Females are more likely to agree that a working knowledge of computers will be necessary in the future and less likely to characterize computers as detrimental to individual privacy. Females also believe that the use of computers as learning aids should begin earlier than do males.

ABSTRACT: Problem Solving Emphasis in a Computer Science Curriculum

Dr. Richard A. Alo', Chairman, Department of Applied Mathematical Sciences, University of Houston-Downtown, Houston, Texas 77002

This paper discusses the design of computer science curricula to meet a student's career goal of entering problem solving/decision making type careers with a Bachelor of Science in computer science. The role of mathematics (discrete and continuous) is discussed along with its interaction with computer science. The nature of a required senior seminar and senior practicum experience is detailed as a significant experience synthesizing a student's training and culminating in a detailed project report emphasizing recognition, formulation and resolution of the project problem.

Dr. Alo' was a member of the Mathematical Association of America's Committee for Undergraduate Programs in Mathematical Sciences and a participant in the 1982 Williams Conference on "Discrete Mathematics in the First Two Years." As a result, he was one of the author's of CUPM's recommendations for undergraduate programs in mathematical sciences (including computer science) and has designed undergraduate programs in mathematical and computer sciences.

RESOURCES FOR SECONDARY COMPUTER SCIENCE TEACHERS

Vicki Smith Bigham, Chair
Consultant, ICCE Past President
Houston, TX

ABSTRACT

Panel members will discuss a variety of resources available for secondary computer science teachers.

The recent developments in computer science teaching on the high school and junior college levels have improved instruction dramatically but have also placed new demands on teachers. Many computer science teachers are undertrained and are desperately seeking sources of information to assist them with their teaching. The editor of the newsletter for the International Council for Computers in Education (ICCE) Special Interest Group for Computer Science (SIGCS) will describe the newsletter as well as journals, teacher's guides, and other materials available for this purpose, with emphasis on those items of immediate practical interest.

The president of SIGCS will address three related topics: the APCS-A Exam, teachers' groups and Dr. Pascal. The APCS-A Examination, offered by the College Board, represents an attempt to encourage more schools to offer introductory programming courses that will be recognized by colleges and universities. The computer science courses that will be accepted as a basis for students' later studies will be discussed. In the fall of 1983, the Friends of 2^7 came into existence to help teachers in the vicinity of Route 128 (=2^7) to teach the APCS course. Few of those teachers had course credits in computer science, much less degrees in the topic. The formation of the group will be described as well as identification of significant topics, software, textbooks, and teaching materials. Dr. Pascal, a product of Visual Software, offers a development system that enables students to see what each statement of a program does. Included in the discussion of this software will be how to assist teachers in discovering the best and latest software before they buy new systems.

The Texas Education Computer Cooperative-funded Pascal Teacher Training Project will be highlighted. This project was designed to assist teachers in presenting the Pascal programming language and to equip them with the methodology appropriate for its delivery in the secondary environment. The project manager will describe the products developed, which include: five videotapes of a ten-session Pascal training program that was delivered over an interactive television fixed service (ITFS) network; a 394-page set of training materials, including outlines of the sessions and transparency masters; and a 557-page curriculum guide, including sample course outlines, hardware/ software configurations, overhead masters, sample problems/solutions, and Apple IIe and Macintosh versions of Pascal problem set data disks.

The audience will be invited to add to the discussion of resources available.

PANELISTS

Tim Corica
The Peddie School
Hightstown, NJ

Zab Warren
Phillips Academy
Andover, MA

USE OF MICROCOMPUTER SOFTWARE IN THE DEVELOPMENT OF MULTIPLE-CHOICE READING TESTS FOR ELEMENTARY AND SECONDARY PUBLIC SCHOOL STUDENTS

Elisabeth G. Flach

Montgomery County Public Schools
850 Hungerford Drive
Rockville, Maryland 20850

ABSTRACT

A process intended to improve the quality and expedite the writing of multiple-choice items for criterion-referenced tests of reading comprehension for elementary and secondary public school students was devised by a large suburban school district. A variety of microcomputer applications software was used to provide both support materials and information in a feedback loop to assure item writer application of both the rules for the selection of discourse from the domain of acceptable reading materials and the grade-level specifications for generating items. Daily feedback of such information assisted in controlling overuse of specific discourse sources and content areas and aided item writers in selecting vocabulary of appropriate difficulty. Use of these procedures helped assure curricular representativeness of test items and a consequent common interpretation of test scores. Main frame programs provided all common descriptive statistics and permitted analysis relative to the procedures presented here.

OBJECTIVES

Primary Objective

A process using a variety of microcomputer applications software to improve the quality and expedite the completion of locally developed multiple-choice test items was developed by a large suburban school district. As the possible applications of such software programs are probably too numerous to list, this paper will outline only one such set of procedures designed to incorporate activities which, though desirable, are often considered too time-consuming and expensive to be undertaken by local school districts. The overall goal for the project was to obtain a product that is clearly superior to that typically achieved.

During initial project planning stages, decisions regarding the content and wording of the curriculum objectives to be tested were made judgmentally on the local level, the result of concern for curricular relevancy, completeness of coverage, developmental appropriateness, etc. Steps in the item-writing process could then be carried out with improved quality and efficiency (Appendix 1).

Secondary Objectives

Secondary objectives involved simplifying all steps in the item-writing process, providing well organized, legible copy for all phases of the project, and storing all necessary documentation on floppy disks. This paper covers only the item-writing activities: The preliminary objectives macrostructure planning and statistical analyses of student responses and subsequent revision processes were reported in other papers, the chief of which was entitled A Methodology for Identifying Sex- and Ethnic-biased Achievement Test Items, presented at the annual meeting of the American Educational Research Association in Boston, Massachusetts, April, 1980 (Flach and Berk, 1980).

THEORETICAL FRAMEWORK

A Theory-Based Process

This test development effort is an ongoing process which is entirely theory-based, the management of which has been under the direction of a full-time development coordinator, R. Ted Schuder, who, along with a great many outside consultants, has overseen the relating of current research to both curriculum and test development (Schuder, 1986).

Development of Linguistic Parameters

These reading test items were written under a strict set of grade-specific generation rules for such linguistic parameters as total number of words required for coherent excerpts suitable as item stems, average sentence length permissible for excerpts, the percentage of words allowed as "uncommon" according to the Dale List of 3000 Familiar Words (Dale and Chall, 1948), etc. (Appendix 2). These rules may be considered analogous to those of Hively et al. (1973) for mathematics items, but were entirely locally developed (Flach and Berk, 1980).

Development of Domain Specifications

Domain specifications were also considered of first importance in order to assure an adequate sampling of the domain of reading materials in use in the schools. Highly detailed surveys were carried out involving not only teacher listing of materials used in classrooms, but also percent use for various student strata including those with high, average, and low ability, those in special programs such as English for Speakers of Other Languages, etc.

TECHNIQUES

Steps in the Process

The steps in the process involved:
(1) organization and analysis of a survey of the curricular domain of the materials to act as sources for test excerpts, whether they be of an expository, narrative, or quantitative nature; (2) development of item specifications including grade-level boundaries for sentence length, vocabulary level, and density of uncommon words through the use of one of several common "readability" programs; (3) the sorting of alphabetized word lists by part of speech as an aid in selecting appropriate vocabulary for directions, questions, and answer options; (4) the building of data bases for the storage and retrieval of various aspects of item content such as objective classification, publisher source, type of topic, passage structure components such as number of characters, use of dialogue, complexity of clauses, etc.; and (5) the printing out and returning of data base information to managers and item writers during workshops to assure refocusing of efforts as required.

Use of Conversion Software

Since the inception of the item-writing project in 1976, the use of software to convert print material from one computer system to another has become standard procedure. For instance, test items entered on an Apple microcomputer were converted with "turnover" software (Vertex Systems, 1984) to "Wordstar" software (MicroPro International Corporation, 1984) and thence to an offset printing system which provides for the printing of various fonts selected as suitable for students in specific grades. Likewise, test items entered on a remote terminal IBM system were also converted through the use of "turnover" software to various fonts as above. In addition, materials typewritten in specific typefaces such as IBM's "Prestige Elite," "Courier 10," or "Letter Gothic" were scanned and converted to an offset system.

Use of Main Frame Programs to Track Item Statistics and Student Demographic Data

In our project, students mark on machine-scorable answer sheets which are processed centrally, with responses stored on main frame magnetic tapes, then merged with main frame stored student demographic information such as ID number, school, grade, age, race, sex, and other information needed for item bias and instructional effect analyses. An item analysis program, "ITEMAN" is then run to analyze student data for percent students selecting options by ability level, race, sex, etc., (Berk and Griesemer, 1976).

All common descriptive statistics are also obtained such as percent students answering each item correctly, reliability statistics for whole-test and for individual subtests (objectives), and item-total and item-subtest correlations. Use of these statistics enables us to select items with the specific characteristics of 60 to 80 percent students selecting the correct option, a high correlation of the item with total test and subtest, balanced selection of wrong options, and lack of bias related to sex and race as revealed by performance levels consonant with subgroup ability levels. Items failing to meet these criteria are revised until criteria are met, or eliminated entirely.

Use of Microcomputer Software to Improve Quality and Expedite Item Completion

Since the beginning of the item-writing project, a great deal of software has become available which makes test development feasible on the local level. Without such software, it was formerly difficult if not impossible to control overuse of the materials of certain publishers, and to assure the use of a representative variety of authors, scenarios, locales, etc. Tracking of the logistics from initial discourse selection through the final printed test was also extremely difficult. Finally, assuring appropriate difficulty through the use of readability formulas and graded vocabularies was time-consuming and often inaccurate. Judicious use of data base and speciality software has made this type of control a reality.

Data Sources (Objectives, Materials, Specifications)

For this paper, materials used to provide examples of these procedures will be those used in the development of criterion-referenced test items in reading comprehension for public school students in grades one through eight in a large mid-Atlantic suburban school district. Included will be sample presentations of textbook source listings, lists of grade-specific objective specifications, item-writer tally sheets for tracking excerpt characteristics, and various data bases and prepared word lists. Included also will be the software titles, publishers, and addresses necessary to create the support materials described here. Samples of these support materials are included as the appendixes of this document, with sources cited in Appendix 1, and addresses and phone numbers listed under References. Complete forms will be available as handouts at the presentation, and a screen search of the vocabulary data base and a readability analysis program will be loaded on two Apple IIe microcomputers.

Educational Importance

The purpose of using various forms created with software to focus on objective, content, and linguistic parameter control was to assure generalizability of performance status on the set of items to a specified domain of tasks and behaviors in both classroom and independent situations. A significant increase in efficiency in developing items with these characteristics was noted when these procedures were applied. Specifically, statistical analysis of piloted items developed by these methods displayed grade-level appropriateness i.e., were answered correctly by a large portion of developmentally able students receiving adequate instruction in grade-level materials. Because of the care in applying linguistic parameters through the use of pre-set boundaries and readability analyses of discourse selected for item stems, students of the targeted ability levels generally performed well on the item sets.

In addition, the use of the information in content and source data bases in a feedback loop helped assure curricular representativeness which in turn helped obviate interactions between student familiarity with discourse sources and item difficulty. Stereotyping and apparent race and sex bias were partially eliminated during the item-writing process by rejecting or restricting inappropriate scenarios in discourse excerpts through item writer education by managers who ran readability and data base entries daily.

To summarize, as the ultimate goal of the use of item-generation rules and representative source and materials domains was to insure a common interpretation of test scores, procedures which aid in fully explicating rules applications through ongoing item writer feedback procedures should serve to improve ultimate test score validity.

REFERENCES

Apple Computer, Inc., c1982, c1983, 20525 Mariani Avenue, Cupertino, California 95014, phone: (408) 996-1010.

Berk, R. A., and Griesemer, H. A. (1976). ITEMAN: An item analysis program for tests, questionnaires and scales. Educational and Psychological Measurement, 36, 189-191.

Dale, E., and Chall, J. S. (1948). A formula for predicting readability. Educational Research Bulletin, 27, 11-20, 37-54.

Flach, E. G., and Berk, R. A. (1980). A methodology for identifying sex- and ethnic-biased achievement test items. Paper presented at the annual meeting of the American Educational Research Association, Boston, Massachusetts.

Harris, A. J., and Jacobson, M. D. (1982). Basic Elementary Reading Vocabularies. New York: Macmillan Publishing Co., Inc.

Hively, W., Maxwell, G., Rabehl, G., Sension, D., and Lundin, S. (1973). Domain referenced curriculum evaluation: A technical handbook and a case study from the MINNEMAST Project (CSE Monograph Series in Evaluation, No. 1). Los Angeles: Center for the Study of Evaluation, University of California.

IOX Basic Skills Word List, Grade 1-12. (1980). Instructional Objectives Exchange, Box 24095, Los Angeles, California 90024.

McCain, R. L., and Flach, E. G. (1987). A pre-pilot feedback process for writing test items for secondary school computer education courses. Paper accepted for presentation at the annual meeting of the National Council on Measurement in Education, Washington, D. C.

MicroPro International Corporation, c1979, c1984, 33 San Pablo Avenue, San Rafael, California 94903, (415) 499-1200.

Funk Software Inc., c1984, P. O. Box 1290, Cambridge, Massachusetts 02238.

Schuder, R. T. (1986). Reconceptualizing reading in a public school system. In J. A. Orasanu (Ed.), A Decade of Reading Research: Implications for Practice. Hillsdale, New Jersey: Lawrence Erlbaum.

Stone, C. R. (1956). "Measuring Difficulty of Primary Reading Material: A Constructive Criticism of Spache's Measure," Elementary School Journal, 57, 36-41.

Stoneware Incorporated, D B Master Associates, c1984, 50 Belvedere Street, San Rafael, California 94901.

Sunburst Communications, c1981 MECC, Minnesota Educational Computing Consortium, 2520 Broadway Drive, St. Paul, Minnesota 55113.

Vertex Systems, Inc., c1984, Suite 3, 6022 West Pico Boulevard, Los Angeles, California 90035, (213) 938-0857.

APPENDIX 1

Steps in the Item-Writing Process

1. Item writer refers to Instructional and Performance Objectives Tested (Appendix 2), selects suitable discourse from surveyed domain as test item stem. Forms prepared with DOS Applewriter 3.3, Apple Computer, Inc., and Sideways, Funk Software, Inc.

2. Item writer refers to Suggested Specifications (Appendix 3), accepts or rejects discourse, writes options, checks vocabulary level of key words. Form prepared with DOS Applewriter 3.3, Apple Computer, Inc., and Sideways, Funk Software, Inc.

3. Item writer enters a unique number and own initials on Initial Tally Sheet (Appendix 4). Unique number charts prepared with ProDOS Applewriter 2.1 using "Replace" command over the series of grade levels. Tally Sheet alphabetized with Appleworks, Apple Computer, Inc.

4. Item writer enters checks on Content Check List (Appendix 5). Form prepared with ProDOS Applewriter 2.1: "Replace" command used to create vertical dividers. Checks entered under the following headings:

   - Structural aspects: number of characters, presence or absence of dialogue, number of speakers, complexity for grade, etc.
   - Genre-theme: e.g., achievement, adventure, adversity, animal, etc.
   - Demographic status: sex, age, ethnicity, socio-economic status
   - Environment: country, locale (suburban, rural, etc.), setting (business, farm, school, service, sports, transportation, etc.)
   - Role of major characters: controls others, practices profession, does job, provides support, etc.

5. Item writer carries out screen search for grade level of difficult words. Data base of words built from four vocabulary lists, grades one through three; from two lists, grades one through eight, etc. (For word lists, see references for Dale and Chall, IOX, Harris and Jacobson, and Stone.) Data base program was D B Master, Stoneware, Inc.

6. Item writer completes Worksheet (Appendix 6); refers to an item review sheet to check format, content, specifications, and logistical procedures. (Shell worksheet blank form created, "saved," then successive grade-level specifications entered as appropriate.) ProDOS Applewriter 2.1 used. (For an example of item writers entering items directly on a microcomputer software shell, see McCain and Flach, 1987).

7. Manager runs discourse passage through Finding Readability Levels, Sunburst Communications, and returns printout to item writer (Appendix 7). Item writer makes revisions as appropriate or discards item.

8. Manager enters data base(s): Data base information may be sorted and returned to item writers (Appendix 8), merged with publisher letters, summarized in calculated fields for average book cost, etc. (Appleworks, Apple Computer, Inc., or D B Master, Stoneware Incorporated).

APPENDIX 2
OBJECTIVES TESTED
(SAMPLE)

Instructional Program in Reading/Language Arts
INSTRUCTIONAL AND PERFORMANCE OBJECTIVES TESTED BY GRADE FROM PROGRAM OF STUDIES
Criterion-Referenced Reading Tests: Comprehension of Narrative Excerpts
End-of-Year Testing 1986

OBJECTIVES Format	Grades/Levels 6/U	7/V	8/W	OBJECTIVES FROM PROGRAM OF STUDIES
CHARACTERIZATION	■	■	■	TO DEVELOP UNDERSTANDING OF CHARACTERS:
Trait	X	X	X	Given a character - identify the character's traits or behaviors
Speaker's Emotion	X	X	X	(R-W) Given speech expressing feeling - identify the character's feeling (V&W) Given a character's thoughts or feelings - tell what the thoughts or feelings reveal about the character
ATTITUDE	■	■	■	TO DEVELOP UNDERSTANDING OR CHARACTER'S ATTITUDE
Character's Attitude	X	X	X	Given speech representing attitude - identify the attitude of the speaker
MOTIVATION	■	■	■	TO DEVELOP UNDERSTANDING OF CHARACTER'S MOTIVATION
Speaker's Intention	X	X	X	(Q) Given speech intended to deceive or convince - identify the speaker's intent (S-W) Given speech representing intent - identify the speaker's intent
Reason for Action	X	X	X	Given a character's action - give reason(s) for the action

APPENDIX 3
SUGGESTED SPECIFICATIONS
(SAMPLE)

INSTRUCTIONAL PROGRAM IN READING/LANGUAGE ARTS
Suggested Specifications for Writing Items for Tests of Narrative Excerpts

Domains Covered		Stem Specifications				Response Specifications		
Grade Level	Discourse Forms [1]	Average Number Words in Sentence	Ratio Uncommon To Common Words [2]	Number of Sentences in Discourse [3]	Total Words in Discourse [3]	Number Answer Options	Number Words Each Sentence	Number Sentences Each Answer Option
6th	Short Narratives Folk Tales Novels Legends Myths	14-17	1/12	1-8	14-136	5	1-17	1-4
7th	Folk Tales Fables Novels Biographies Autobiographies Legends Short Stories	15-22	1/9	1-9	15-198	5	1-22	1-4
8th	Novels Biographies Autobiographies Legends Myths Short Stories	15-24	1/8	1-10	15-240	5	1-24	1-4

GRADE TWO: INITIAL TALLY SHEET

2-Q

INSTRUCTIONAL PROGRAM IN READING/LANGUAGE ARTS
WORKSHOP 2055 SUMMER 1984
BOOKLISTS: GRADE TWO

GRADE TWO: BASALS

Publisher	Title	OBJECTIVES CHARC	MOTIV		CONFL		EVENT	VOCAB		DIALG	COHES		SETNG	
		Trait	Speaker Intent	Reason Action	Problem	Resolution	Next Event	Synonym	Cloze	Who Talk?	Temporal	Causal Rels	Temporal	Locative
Addison-Wesley 1982	The Nitty Gritty, Rather Pretty City													
	The Nitty Gritty, Rather Pretty City													
Ginn & Co. 1984	Glad to Meet You (Ginn Reading)													
	Give Me a Clue (Ginn Reading)													
Harcourt Brace 1983	World of Surprises (HBJ Reading)													
	People and Places (HBJ Reading)													
	Heart of the Woods (Odyssey Lit)													
Heath & Co. 1983	Marching Along (American Readers)													
	Turning Corners (American Readers)													
Holt Rinehart 1983	People Need People (BRSH: Series)													
	Way of the World (BRSH Series)													
Houghton Mifflin Co. 1983	Skylights (Houghton Mifflin Reading)													
	Towers (Houghton Mifflin Reading)													

APPENDIX 4: SAMPLE INITIAL TALLY SHEET

APPENDIX 5

SAMPLE DISCOURSE CONTENT CHECK LIST

Item Writer______________________Page No.____

(Enter Unique Item Number)

										Total
STRUCTURAL ASPECTS										
NUMBER CHARACTERS										
Unstated										
One										
Two										
Three or More										
DIALOGUE										
Present										
Absent										
NUMBER SPEAKERS										
One										
Two										
Three or More										
COMPLEXITY FOR GRADE										
Simple										
Average										
Complex										
CONTENT ASPECTS										
GENRE-THEME										
Achievement										
Adventure										
Adversity										
Animal										
Career										
Competition										
Daily Life										
Discovery										
Family Relationships										
Fantasy (Science Fiction)										
Friendship										
History										
Horror										
Justice										
Mystery										
Nature										
Personal Reaction										
Responsibility										
Success										
School										
Other (List)										

UNIQUE NO. X-110

APPENDIX 6: SAMPLE COMPLETED WORKSHEET: GRADE EIGHT

Instructional Program in Reading/Language Arts ITEM WRITER: L. Zerfing
Workshop 2055: Item Writing--Narration
SUMMER 1984

Publisher: Scott Foresman

Series: Images

Title: "Haunted Summer"

Author: Hope Dahl Jordan

Copyright Date: 1977 Page(s) p. 223 Book II

GRADE EIGHT OBJECTIVES--LEVEL W

CHARC
__Trait
__Speaker's Emotion
__Speaker's Attitude

MOTIV
__Speaker's Intention
__Reason for Action

CONFL
X Problem

EVENTS
__Next Event

VOCAB
__Synonym
__Cloze

DISCOURSE:

Number Sentences in Discourse: (1-10) 10
Range Number Words in Sentences: (15-24) 4-22
Total Number Words in Discourse: (15-240) 111
Ratio Uncommon to Common Words: (1/8) 1-9

OPTIONS:

Range Number Words in Sentences: (1-24) 13-15
Ratio Uncommon to Common Words: (1/8) 1-7

BEGIN DISCOURSE: "Seth had already..."

END DISCOURSE: "you by asking questions"

CHANGES TO DISCOURSE:

ANSWER OPTIONS: What is Seth's problem?

OPTION TYPE:

A. (✓) Seth is upset by a secret that is having a negative effect on his life. — summary

B. Seth is worried that his poor grades will prevent him from being accepted into college — partial

C. Seth spends too much time telling jokes rather than concentrating on his school work. — inappropriate extension

D. Seth feels compelled to change the subject when any conversation focuses on him. — literal

E. Seth is angry with his father for making him resign from his job — partial

FORMAT CHECKS:

CONTENT CHECKS:

APPENDIX 7

DEATH IN SAMARRA[1]

PASSAGE W-41

THERE WAS A MERCHANT IN #BAGDAD WHO SENT HIS SERVANT TO MARKET TO BUY PROVISIONS AND IN A LITTLE WHILE THE SERVANT CAME BACK, WHITE AND TREMBLING, AND SAID, MASTER, JUST NOW WHEN I WAS IN THE MARKETPLACE I WAS JOSTLED BY A WOMAN IN THE CROWD AND WHEN I TURNED I SAW IT WAS DEATH THAT JOSTLED ME. SHE LOOKED AT ME AND MADE A THREATENING GESTURE; NOW LEND ME YOUR HORSE, AND I WILL RIDE AWAY FROM THIS CITY AND AVOID MY FATE. I WILL GO TO #SAMARRA AND THERE DEATH WILL NOT FIND ME." THE MERCHANT LENT HIM HIS HORSE, AND THE SERVANT MOUNTED IT, AND HE DUG HIS SPURS IN ITS FLANKS AND AS FAST AS THE HORSE COULD GALLOP HE WENT. THEN THE MERCHANT WENT DOWN TO THE MARKETPLACE AND HE SAW DEATH STANDING IN THE CROWD AND HE CAME TO DEATH AND SAID, "WHY DID YOU MAKE A THREATENING GESTURE TO MY SERVANT WHEN YOU SAW HIM THIS MORNING?" "THAT WAS NOT A THREATENING GESTURE," DEATH SAID, "IT WAS ONLY A START OF SURPRISE. I WAS ASTONISHED TO SEE HIM IN #BAGDAD, FOR I HAD AN APPOINTMENT WITH HIM TONIGHT IN #SAMARRA."

NUMBER OF SENTENCES:	7
NUMBER OF WORDS:	198
NUMBER OF SYLLABLES:	242
WORDS OF 6 OR MORE LETTERS:	37
3 OR MORE SYLLABLE WORDS:	7
% OF 3 OR MORE SYLLABLE WORDS:	3.5
AVERAGE SENTENCE LENGTH:	28.3
AVERAGE LETTERS PER WORD:	3.9
AVERAGE SYLLABLES PER WORD:	1.2

THERE ARE 27 WORDS NOT IN THE DALE-CHALL WORD LIST. CORRECTED DALE-CHALL GRADE LEVEL: 9TH-10TH GRADE.

THE FRY GRADE LEVEL IS 8.

THE FLESCH READABILITY SCORE IS 74.7. THIS SAMPLE IS AT THE 7TH GRADE LEVEL.

OPTIONS-SAMARRA

PASSAGE OP-W41

HE WILL DIE ON HIS WAY TO #SAMARRA.
HE WILL MEET HIS DEATH THAT NIGHT.
HE WILL RETURN FROM #SAMARRA SAFELY.
HE WILL RIDE THE HORSE TO ITS DEATH.
HE WILL AVOID DEATH BY GOING TO ANOTHER CITY.

THERE ARE 3 WORDS NOT IN THE DALE-CHALL WORD LIST. CORRECTED DALE-CHALL GRADE LEVEL: 5TH-6TH GRADE.

THE FRY GRADE LEVEL IS 2.

THE FLESCH READABILITY SCORE IS 92.3. THIS SAMPLE IS BELOW THE 7TH GRADE LEVEL.

Grade Level of Words on Two Word Lists

	Harris-Jacobson (1982)	IOX (1980)
die	2	3
meet	2	3
death	3	5
return	2	2
ride	P	1
horse	1	1
avoid	4	4
another	1	1
city	Q	2

---

[1] from Maugham, S. An Appointment in Samarra. In P. McFarland, F. Feagin, Hay, S. Liu, F. McLaughlin & N. Willson (Eds.) Focus on Literature: PEOPLE. Boston: Houghton Mifflin Company, 1981, 1978, p. 61.

APPENDIX 8

SAMPLE DATA BASE: ALL INFORMATION: GRADE 3

```
                         ALL INFORMATION                        PAGE
1

        ***** INSTRUCTIONAL PROGRAM IN READING/LANGUAGE ARTS *****
    *** CRITERION-REFERENCED TEST ITEM INFORMATION: NARRATION  C1984 ***
------------------------------------------------------------------------
-----

    Q0005
    GRADE:  3  CHARC:  TRAIT                       STATUS:  OLD TEST
    SELECTION:  MISSISSIPPI POSSUM                   PAGE(S):  22
    BOOK TITLE:  WITH SKIES AND WINGS                COPYRIGHT:  1969
    AUTHOR/ED:  MILES
    PUBLISHER:  GINN
    SCENE:  POSSUM AFRAID OF MUCH         KEY:  SCARED
    MACROSTR:                            INFER:     WRITER:

    Q0008
    GRADE:  3  CHARC:  TRAIT                       STATUS:  OLD TEST
    SELECTION:  MR. MOONLIGHT AND OMAR               PAGE(S):  78
    BOOK TITLE:  WITH SKIES AND WINGS                COPYRIGHT:  1969
    AUTHOR/ED:  HOLDING
    PUBLISHER:  GINN
    SCENE:  CAMEL DRIVER BEATING CAMEL    KEY:  IMPATIENT
    MACROSTR:                            INFER:     WRITER:

    Q0011
    GRADE:  3  EVENT:  NEXT EVENT                  STATUS:  OLD TEST
    SELECTION:  FRECKLE JUICE                        PAGE(S):  17
    BOOK TITLE:  WINDCHIMES                          COPYRIGHT:  1977
    AUTHOR/ED:  BLUME
    PUBLISHER:  HOUGHTON MIFFLIN
    SCENE:  GIRL THROWS PAPER TO BOY      KEY:  COUNTED
    MACROSTR:                            INFER:     WRITER:
```

SAMPLE DATA BASE: AGE, ENVIRONMENT, SETTING, ROLES, GENRE, TOPIC

File: LEVEL 0
Report: Listing3

Page 1
07/29/84

Unique No	Age	Environment	Setting	Roles	Genre	Topic
2-161	child	outdoors	baseball	players	s n	winning
2-127	child	school	classroom	teacher, class	s n	winning
2-044	child	uns	home	killer, mother, giant	folk tale	triumph over evil
2-123	child, adult	uns	uns	helper, adult	s n	helping
2-102	child	outdoors	beside dike	hero	folk tale	hero saves day
2-106	child, uns	outdoors	dragon's lair	controller, victim	folk tale	slay dragons
2-010	uns	outdoors	uns	teller, victim, helper	animal story	help distressed
2-045	child, uns	outdoors	home, river	pet lover, pet	s n	giving up pet
2-242	uns	home	bedroom	sleeper, wakener	s n	wake up sleepyhead
2-006	uns, adult	outdoors	hot dog stand	trickster, owner	s n	trickery
2-186	adult, all	outdoors	doorway of house	victim, helpers	folk tale	help distressed
2-185	child, aged	outdoors	little house	helper, infirm	s n	helping elderly
2-166	child, adult	outdoors	road	helper, observer, pet	s n	helping pet
2-065	child, adult	school	classroom, playground	student, teacher	s n	shyness
2-042	child	home	living, bedroom	helper	s n	helping plant
2-141	uns	outdoors	vegetation	eater	s n	basic needs

A CUSTOMIZED AUTHORING SYSTEM FOR RAPIDLY PROGRAMMING BASIC COMPETENCIES

K. Wayne Goode
Bernard J. Schroer
Johnson Research Center
University of Alabama in Huntsville
Huntsville, Alabama 35899

ABSTRACT

This paper discusses a series of software development tools for writing remediation software for the Apple microcomputer. These tools include an authoring shell, a set of machine language utilities and a rapid graphics development package. To date over 80 mathematics competency programs have been developed using these development tools.

INTRODUCTION

In 1984 the Alabama Legislature funded the University of Alabama in Huntsville (UAH) and the University of North Alabama (UNA) to establish the Alabama Educational Computing Research and Development Network. The Network's task is to develop computer software corresponding to the basic competencies identified by the Alabama Department of Education (Dept of Ed, 1979).

The Alabama Department of Education has identified 199 competencies in mathematics, reading and language arts. Tests are administered in the third and sixth and ninth grades. If a child fails to pass a competency the teacher is required to provide remedial instruction to the student and to document the remedial instruction activities.

The development of a competency software program consists of five phases: specification definition, software development, software verification, pilot school evaluation and software distribution. UNA selects a competency and prepares a detailed specification using task analysis and storyboarding. UAH writes the software and returns the program to UNA for review. Undergraduate students are being used to write the programs. Once the program has completed the review process a teacher's guide is prepared and the software sent to fourteen pilot schools for field testing. Upon completion of the review by the pilot schools the software is ready for distribution statewide.

SOFTWARE DEVELOPMENT TOOLS

Since the project requires a large number of competency programs to be written rapidly and correctly, the first task was to search for commercially available software development tools. The first development tool to be evaluated was SuperPILOT. SuperPILOT is an easy system to use; however, it is oriented to much smaller programs than required by the Network. PILOT was originally written as an authoring system for teachers who did not want to learn to program. The oversimplification of PILOT limits the power of the program. Even though SuperPILOT is a revised and extended version of PILOT it is still limited. Graphics can be used; however, they are limited. Characters can be edited to create objects, but their size is limited. To use a larger pre-drawn picture requires that each picture be saved in a separate file. The picture can only be one size: the full screen. This makes animation impossible. Also, PILOT is slow since it is written in Pascal instead of machine language.

Several high level languages were also evaluated; however, such languages as C and Pascal are not readily available for the Apple and do have the necessary graphics routines. Therefore the decision was made to develop our own tools for writing the competency software: GAS - Authoring shell; GAP - Ampersand Program; SNAP - Sketch Network Arrangement Program and MSF - Make Start Frames. This paper discusses each of these tools.

GAS - AUTHORING SHELL (UAH, 1986)

All the competency programs have the same structure and are divided into the following sections: Introduction, Pretest, Instruction, Practice, Posttest, End, and Teacher's Menu.

The introduction displays the title screens which state the competency the program covers and the grade level. It asks the user for his name so that it can be used for feedback in the program. The pretest consists of four questions. If the student answers three or four questions correctly the program ends. If the student fails the pretest, the instruction program is executed. After the student finishes the instruction program, the practice program is executed.

The practice test consists of twenty questions. If the student answers the first five questions correctly, then the program ends. Otherwise, the student is given up to fifteen more questions. After the tenth question, the student is given the option of answering practice problem or going to the posttest. The posttest is similar to the pretest. The student is given four questions. If the student answers three or four questions correctly the program ends. If the student fails the posttest, the student goes back to the instruction program.

The teacher's menu is a menu program that can be used by the teacher to select a particular program: pretest, instructions, practice, posttest or return to program. To run the menu control-A is pressed whenever the space bar is displayed.

The GAS Authoring Shell was written because each of the competency programs follows the same format. GAS is written in Applesoft BASIC and has the basic program structure of each of the above sections. Included are routines for tasks that need to be done frequently. Some of these routines are described in the following sections.

Press Space Bar

This routine prints "press space bar" and waits. Instead of pressing the space bar the users can use the following additional keys to move to other parts of the program:

Ctrl A	Go to Teacher Menu
Ctrl B	Go back a screen
Ctrl 2	Asks for a screen number and goes to it

Printing Fractions

Routines for drawing fractions in each font are included. These routines print the fraction (T = top and B = bottom of fraction) at the current hi-res cursor location and position the cursor at the right of the fraction.

Move Text

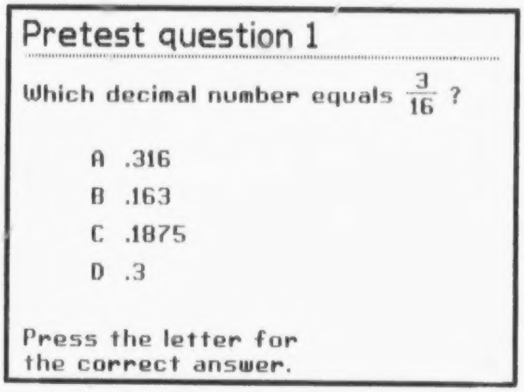
This routine will move a string of text from one location on the screen to another. This routine is useful for animating text.

Multiple Choice Question Routine

This routine prints the choices and prompts the user for an answer. Then the program tells the student if the answer was correct and keeps track of the number of correct answers. Figure 1 shows a listing of the commands to create a sample pretest screen. This program uses the multiple choice question routine, fraction routine, and press space bar routine. (See Figures 2 and 3.)

```
1000  &  AT 0,30
1010  PRINT "Which decimal number equals ";
1020 T = 3:B = 16: GOSUB 600: PRINT " ?"
1030 A1$ = ".316"
1040 A2$ = ".163"
1050 A3$ = ".1875"
1060 A4$ = ".3"
1070 K = 3: GOTO 21000
```

Figure 1. Listing for sample pretest



Pretest question 1

Which decimal number equals $\frac{3}{16}$?

A .316

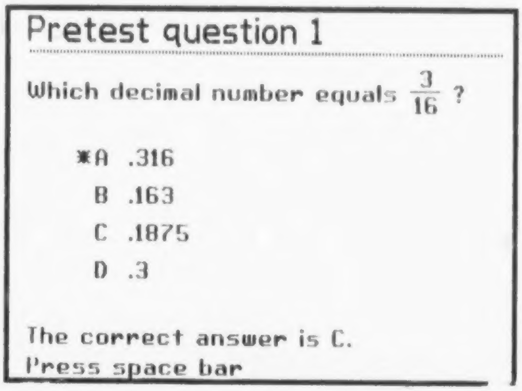
B .163

C .1875

D .3

Press the letter for
the correct answer.

Figure 2. Sample pretest question screen 1



Pretest question 1

Which decimal number equals $\frac{3}{16}$?

*A .316

B .163

C .1875

D .3

The correct answer is C.
Press space bar

Figure 3. Sample pretest question screen 2

One Character Input

This routine displays a flashing question mark at the current hi-res cursor position and inputs a character.

One Digit Input

This routine uses the one character input routine to obtain a number and prints the character.

Two Digit Input

This routine uses the one character input routine to input a one or two digit number and prints it. To input a one digit number press space or return for the second digit. Backspace can be used to erase the first digit.

Advance Subscreen

This routine prints a header for the next subscreen. For example, if the program is at screen 1 then the first time it is called it changes Instruction 1 to Instruction 1a. The next time it is called it changes it to 1b, then 1c, etc.

GAP - AMPERSAND PROGRAM (UAH, 1985)

Through our experience writing software, we have found that BASIC alone was not sufficient for writing graphic intensive educational software. Therefore, GAP was written to address the deficiencies of BASIC. GAP is written in 6502 machine language, is 3K long and adds 52 commands to BASIC. All the commands start with an &.

Table 1 shows these commands. A few notes about syntax structure: Upper case is for commands and lower case for variables. Commands in brackets [ ] are optional. When there are several options they are separated by a /. One of the options is to be used. A [$] means the number can be specified in hexidecimal by putting a $ in front of the number. Additional data can be specified if the command ends with (...). Using these commands, text can be printed to the hi-res graphics screen, underlined or inversed. Figure 4 shows the commands needed to do this and Figure 5 shows the display that results.

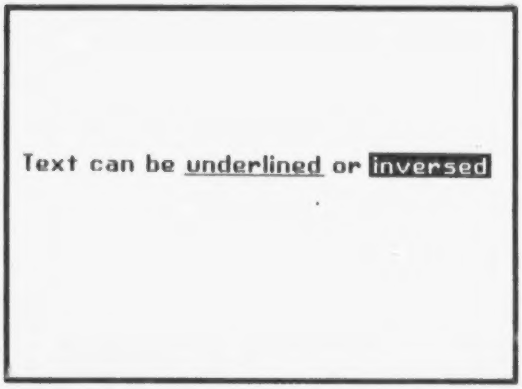
Animation can be done with these commands. Figure 6 shows a program that draws a window with cross bars and animates a ship. By using the & DIM command, only the part of the ship that is visible through the window is drawn. Figure 7 shows the display.

Table I. GAP commands

& ON	turns page on	& HCOLOR = [c1,c2 / INVERSE]	set additional hi-res colors
& CLEAR [p] [,c]	clear page	& FLASH [x1,y1 TO x2,y2]	color area
& STORE	copy page	& INVERSE [x1,y1 TO x2,y2]	inverse area
& GR [n]	page flip	& SCRN (x,y) ,a [,b]	look at screen dot
& PLOT n [AT x,y]	plot picture	& DRAW x1,y1	fill shape
& PLOT =	equal plot	& IF & NOT	IF THEN ELSE statement
& PLOT OR	or plot	& DATA n$	store string
& PLOT XDRAW	xor plot	& READ n$	read string
& PLOT NOT	blank plot	& ASC a$ I n	get a character
& DIM [x1,y1 TO x2,y2]	set graphics window	& ABS a$ I n	reset keyboard and get a character
& DEF [$] n [,f]	picture table/font start	& WAIT n	wait
& AT x,y	position hi-res cursor	& WAIT ON I NOT	enable/disable wait
& STEP x,y	move hi-res cursor	& PDL n,a	read paddle switch
& POS x,y	find hi-res cursor	& POP p,d [... p,d]	play music
& SAVE [n]	save hi-res cursor	& POKE [$] a, [$] n ...	one/two byte poke
& LOAD [n]	load hi-res cursor	& PEEK a,n	two byte peek
& GET n	get font	& [$] n	two byte peek
& SAVE FN	save font information	& CALL [$] n	call
& LOAD FN	load font information	& LOMEM: [$] n	set lomem
& VTAB n [,f]	vertical spacing	& HIMEM: [$] n	set himem
& HTAB n	horizontal spacing	& GOTO [n]	goto
& PRINT HGR / TEXT	print select	& RESUME	return from onerr
& PRINT DRAW / XDRAW	print select	& POP	clear all gosubs and loops
& HPLOT [x,y] [to x1,y1] ...	HPLOT	& FRE	print free space
& SCALE = [n]	set plot scale	&	print memory pointers
		& *	enter monitor
		& REM	remarks

```
100  &  CLEAR 1: &  PRINT  HGR
110  &  GET 2: &  AT 0,40
120  PRINT "Text can be ";: &  POS X1,Y
130  PRINT "underlined";: &  POS X2,Y
140  PRINT " or ";: &  POS X3,Y
150  PRINT "inversed";: &  POS X4,Y
160  HCOLOR= 3
170  HPLOT X1 - 2,Y + 15 TO X2,Y + 15
180  &  INVERSE X3 - 3,Y + 3 TO X4,Y + 15
```

Figure 4. Listing for hi-res text



Text can be underlined or inversed

Figure 5. Hi-res text display

```
100  PRINT ">BLOAD^SHIP,A$A000": &  DEF $A000
110  &  CLEAR 1: &  CLEAR 2: &  GR
120  HCOLOR= 3: &  FLASH 81,50 TO 198,150
130  HCOLOR= 0: &  FLASH 84,53 TO 195,147
140  HCOLOR= 3: &  SCALE= 3
150  &  GR : &  STORE 1
160  &  DIM 84,53 TO 195,147
170  FOR X =  - 178 TO 200 STEP 14
180  &  PLOT 1 AT X + 700 * (X < 0),65
190  &  HPLOT 84,100 TO 195,100
200  &  HPLOT 140,53 TO 140,147
210  &  GR : NEXT
```

Figure 6. Listing for animation

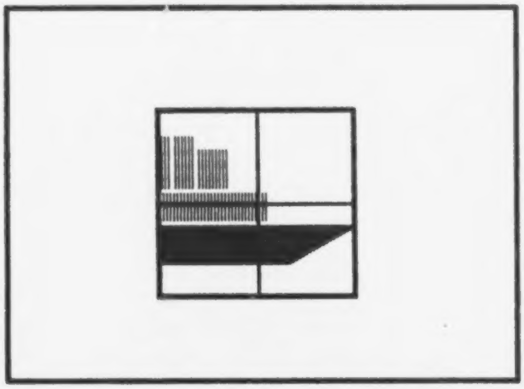


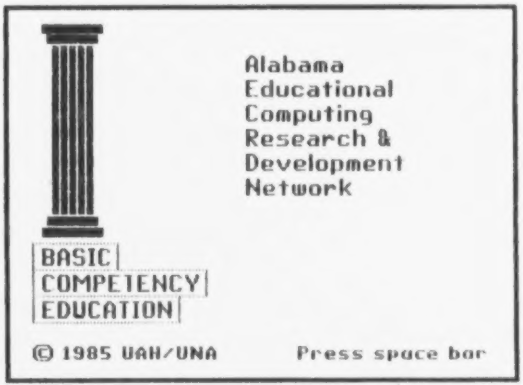
Figure 7. Animation display

SKETCH NETWORK ARRANGEMENT PROGRAM

SNAP (Sketch Network Arrangement Program) makes and edits picture tables from hi-res screens. The programmer can use any drawing program such as Mousepaint or Micro Illustrater to draw the pictures SNAP is used to make the picture table and the GAP commands are used to draw the pictures at any location on the screen.

MAKE START FRAMES

MSF (Make Start Frames) is a special version of SNAP that creates the introduction and credit screens for each program. Four of these screens are the same for all programs. One screen that describes the competency is different in each program. MSF prompts for the information needed for this screen, creates it, and then makes a picture table of the five screens. All of these screens use less space the one screen saved in standard format. SuperPILOT and most other programs save screens in the standard format. Figures 8 and 9 show the first two screens in this picture table.



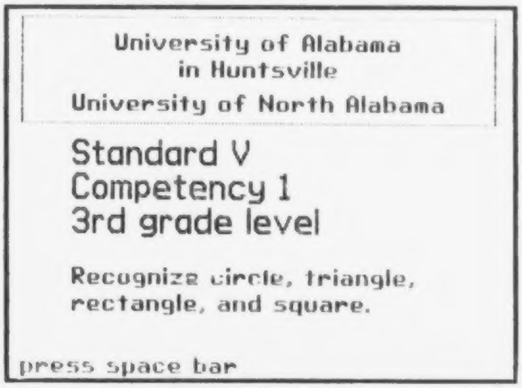
Alabama
Educational
Computing
Research &
Development
Network

BASIC
COMPETENCY
EDUCATION

© 1985 UAH/UNA

Press space bar

Figure 8. Introduction screen 1



University of Alabama
in Huntsville
University of North Alabama

Standard V
Competency 1
3rd grade level

Recognize circle, triangle, rectangle, and square.

press space bar

Figure 9. Introduction screen 2

PROGRESS

A total of 74 mathematics competency programs have been written and are being distributed statewide. In addition, 22 mathematics programs are in some stage of completion. Of those programs being distributed 25 are third grade, 23 sixth grade, and 26 ninth grade.

Five copy sites have been established throughout the state for schools to copy the software. During the first year of distribution over 100 schools have visited one of these copy sites. The schools are permitted to copy the software for distribution within their school systems. Therefore, the number of schools using the software is considerably larger. The schools visiting the copy sites represented 29 of the 67 county and 29 of the city school systems. This represents a 45 percent penetration level. Other users of the competency software are the adult education classes within the Alabama Department of Corrections and the Georgia State Department of Education which has licensed 30 of the programs for statewide distribution.

Another use of the competency software has been as initial learning experiences. For example, a competency program for a third grade student is an initial learning experience for a first or second grader. Likewise, the program could be for advanced learning for a preschooler.

By using the customized software development tools:

- 120 man-hours are required to write and debug a competency program.
- 60 man-hours are required to correct and modify a program after UNA and pilot school review.

CONCLUSIONS

In summary the following conclusions can be made concerning the customized authoring system:

- Rapid development of competency software has been achieved (even with programs requiring complex animation).
- The competency software is being written by undergraduate students with minimal programming experience.
- Minimal training is required to teach the authoring system to new programmers.
- The competency software is being used in Alabama by over fifty percent of the schools.
- The development tools have permitted writing more complex programs than otherwise could have been developed.
- Programs written with the development tools look and work similar so all of the programs are consistent and uniform.

REFERENCES

1. Alabama State Department of Education, *Minimum Standards and Competencies (Reading,Language, and Mathematics) for Alabama Schools*, 1979.

2. Apple Computer, Inc., *Apple SuperPILOT Language Reference Manual*, 1982.

3. Johnson Research Center, *GAP Utility Program Description*, UAH Research Report #445, 1985.

4. Johnson Research Center, *GAS Authoring System*, UAH Research Report #450, 1986.

INSTRUCTIONAL FEEDBACK AND EDUCATIONAL SOFTWARE

Michael Land, PhD
2840 Sheridan Rd.
Department of Computer Education
National College of Education
Evanston, Il. 60201
(312) 256-5150

ABSTRACT

Techniques for effective feedback strategies with computer assisted instruction have begun to emerge from recent research, although gaps still exist in the research base. Of particular concern to educators is the fact that still only a small number of educational software packages employ effective feedback techniques. By NOT employing appropriate feedback techniques, we are losing one of the main advantages of computer assisted instruction.

Introduction

In this presentation, I will discuss the present state of the research on feedback with computer assisted instruction. In addition, I will point out the most effective feedback techniques supported by current research and give concrete examples of feedback messages appropriate for various computer assisted instruction lessons.

I wrote this paper based on my experiences as a computer education instructor, a researcher, a software author, and a specialist in instruction who teaches microcomputer authoring and software evaluation & selection to preservice and inservice teachers. It is my observation that instructional feedback may be the most instructionally significant and yet the least understood and most misused feature of computer assisted instructional design.

Hundreds of pieces of commercial educational software are placed on the market every year. Of these packages, only a tiny portion of them use effective student feedback strategies. Typically, they either ignore feedback or employ techniques not supported by research. Ironically, one of the strengths of instruction delivered by computer is the potential ability to manage effective feedback techniques. To take advantage of this potential, however, those individuals responsible for authoring educational software must be aware of feedback techniques supported by empirical research. Because this is typically NOT the case, it becomes even more important that the buyer or user of software be knowledgeable of current research results concerning the effects of feedback.

What is feedback?

In this paper, I use the term feedback to describe any procedure used to tell a learner that an instructional response is correct or incorrect. This is consistent with Kulhavy (1977), who suggested that in treating the broad question of how postresponse information influences learning, it may be more appropriate to study the common attributes of feedback rather than unique attributes. Treating feedback as a unitary variable, one can present it as ranging along a continuum from a simple "yes, no" to the presentation of additional information (Figure 1).

Conceptually, the simplest type of feedback takes the form of "yes, no". Variations of this explicit value judgement include such comments as "correct, incorrect", "right, wrong", etc. Based upon type, feedback usually falls into one of the following categories (Barringer & Gholson): (1) verbal; () symbolic, such as a tone or a graphic; or (3) a tangible object, such as candy, money, or a token.

Combinations of feedback, defined by the type of student response (right or wrong) they follow, include (1) right-wrong [feedback after right and after wrong student responses]; (2) right-none [feedback after right but none after wrong responses]; and (3) wrong-none [feedback after wrong but none after right responses]. Feedback can also be (1) immediate, or (2) delayed; the delay may range from seconds to hours to days. Feedback can follow after all responses of a particular category (right, wrong, both), or after ratios of particular categories. It may occur during the lesson or at the end of the lesson, or both.

Toward the other end of the continuum (Figure 1), feedback may take the implicit value judgement form of presenting corrective information. Carried further, the feedback can constitute new lesson information. One can visualize feedback, then, as being either positive (following correct responses) or negative (following incorrect responses) and potentially ranging from a simple signal or word to a complex verbal or written communication (Kowitz & Smith, 1985). The primary function of feedback, though, is to locate errors and provide information so the learner can correct them (Cohen, 1985).

Feedback: reinforcing, informational, attention-directing?

There are three main theoretical frameworks attempting to explain how feedback affects achievement. Feedback can be viewed as reinforcing; as informational or corrective; or as attention-directing. The work of Roper (1977) [information feedback using error correction]; Robin (1978) [feedback by proctors] and Bardwell (1981) [written feedback on a quiz] [both Robin and Bardwell studied feedback as reinforcement]; and Lasoff (1981) [phrasing of the feedback message] lend support to the corrective feedback theory. However, in spite of these findings, there is still considerable sentiment in the literature to treat feedback as reinforcing. Anderson, Kulhavy, & Andre (1971) presented feedback to college students either only after correct or only after incorrect responses. Within the context of reinforcement theory, one might hypothesize that the greatest effect would have been on correct responses. In fact, there were no significant differences in the number of errors made or on the posttest.

If feedback is reinforcing, then one would expect that immediate feedback would be more effective than delayed feedback. Among studies indicating this is not the case are those of Kulhavy & Anderson (1972) and Surber & Anderson (1975) indicating that students perform just as well when feedback is delayed a day as when it is given immediately after a response.

The preponderance of current evidence, then, seems to indicate that feedback acts more as a corrective agent rather than as reinforcement.

Research results

Whether studying feedback in a formal or informal manner, one should ask the following questions:

1. Is the feedback immediate or delayed?
2. Is it verbal, symbolic, other, or a combination?
3. Is it provided after all responses (correct and incorrect); after only correct responses; or after only incorrect responses?
4. Does it follow after each response of the type selected, or only after a percentage or number of those responses?
5. Does the message include knowledge of results [KR] such as "right',"wrong, "incorrect"; knowledge of correct results [KCR] (indicates which response from a range of choices is correct); error correction (informs why a response is correct or incorrect; or combinations of these basic types?

Verbal, symbolic, tangible

Verbal and symbolic feedback produce approximately equivalent results that are superior to that produced by tangible feedback. The poorest combination appears to be that of tangible feedback given in the right-none combination (Barringer & Gholson, 1979).

Immediate or delayed?

A close look at Table 1 shows no clear indication of whether immediate feedback or delayed feedback is best for rote or knowledge level learning. Four of the studies in Table 1 (Arnett, 1985; Beaulieu, 1985; Gaynor, 1981; Sales, 1985) reporting computer assisted learning at the knowledge level, reported no significant difference between immediate and delayed feedback. Roper (1977) reported a significant effect for delayed feedback and Gaynor (1981) reported a negative effect for delayed feedback.

With high level learning, however, it seems clear that delayed feedback is more effective than immediate feedback. Gaynor (1981), for instance, reported that feedback given at end-of-session produced better results with high achieving students as contrasted with immediate feedback. She also claimed that immediate feedback was more effective with students working at the knowledge level while end-of-session feedback appeared to be more effective at higher cognitive levels.

Right-wrong, right-none, wrong-none (error identification)

Compared with positive feedback (feedback after correct responses), negative feedback (feedback after incorrect responses) is generally more effective and is more resistant to extinction (Barringer & Gholson, 1979). Students apparently interpret the absence of feedback [after a correct response] as positive feedback. "Unless I'm told it's wrong, I assume it's correct" seems to be the rationale the majority of students use.

The difficulty with providing feedback only after correct responses, such as the study by Sales (1985), is that students are not provided with any type of error detection whereby they can correct those errors. By providing feedback on both correct and incorrect responses, one runs the risk of distracting the student's attention from the lesson with too many interruptions in the lesson.

Knowledge of results vs informational feedback

Table 1 indicates that error detection (informational) feedback is apparently more effective than no feedback (Anderson, 1985;

Anderson, 1985; Bumgarner, 1984; Gilman, 1969; Roper, 1977; and Sfondillas, 1985). Arnett (1985) reported no significant differences. In addition, informational feedback is at least as effective as (Armour-Thomas, 1984); (Land, 1986) or more effective than (Bumgarner, 1984; Collins, 1984; and Noonan, 1984) knowledge of results ("incorrect", "wrong", right") feedback in computer assisted instruction. Informational feedback appears to be most effective when it follows a wrong response. It really is not necessary after a correct response, however, unless one wants to present new information that the student is expected to remember.

Conclusion

Feedback should not be equated with reinforcement; reinforcement helps in generating new behaviors; feedback is a signal that a response is appropriate or inappropriate. The function of feedback is to provide information related to error identification with the idea that the error will be corrected.

Immediate feedback appears to be most effective with the initial acquisition of a concept or rule or with students experiencing difficulty with the subject matter. Conversely, immediate feedback may interfere with the learning of high achieving or confident learners because it slows the lesson - it "gets in the way of" the lesson. Delayed feedback is apparently most effective with higher cognitive skills and with more confident or higher achieving students. It appears to facilitate retention better than immediate feedback.

Informational feedback is generally more effective than no feedback and knowledge of results feedback, but is most effective following incorrect responses [wrong-none]. Roper (1977) recommended immediate feedback for students with no prior knowledge of the lesson contents, but recommended immediate knowledge of results ["correct", "incorrect"] with delayed informational feedback for students with prior knowledge of the lesson contents.

The feedback message itself should detract as little as possible from the main lesson. The message should be appropriate, simple, clear, and concise, particularly with delayed feedback.

Table 1 indicates a gap in our knowledge concerning feedback and computer assisted learning. However, there is an even greater gap between what we know about feedback and the feedback techniques currently reflected in the majority of educational software. We have the knowledge and the potential to produce software with much more effective feedback techniques but we as software users will have to insist on better feedback techniques in educational software it we expect to get it.

References Cited

Anderson, R. C., Kulhavy, R. W., & Andre, T. (1971). Feedback procedures in programmed instruction. **Journal of Educational Research, 62(2)**, 148-156.

Anderson, J. A. (1985). Effects of concurrent confidency and correctness of response feedback provided during computer assisted instruction. **DAI, 46/07-A**, Doctoral dissertation, University of Oregon.

Armour-Thomas, E. (1984). Microcomputer teaching concepts: Types of computer feedback in learning of relationship concepts at kindergarten level. **DAI, 46/03A**, Doctoral dissertation, Columbia University.

Arnett, P. P. (1985). Effects of feedback placement and completeness within Gagne's model for computer assisted instruction lesson development on concept and rule learning. **DAI, 46/09A**, Doctoral dissertation, University of Georgia.

Bardwell, P. P. (1985). Feedback: How does it function? **Journal of Experimental Education**, 4-9.

Barringer, C. & Gholson, B. (1979). Effects of type and combination of feedback upon conceptual learning by children: Implications for research in academic learning. **Review of Educational Research, 49(3)**, 459-478.

Beaulieu, J. E. (1985). A study of the effects of selected feedback delay intervals upon retention of science material in a computer assisted instructional task with junior high school students. **DAI, 46/07A**, Doctoral dissertation, University of Oregon.

Bumgarner, K. M. (1984). Effects of informational feedback and social reinforcement on elementary students' achievement during CAI drill and practice on multiplication facts. **DAI, 45/04A**, Doctoral dissertation, Seattle University.

Buss, A. H., Weiner, M., & Buss, E. (1954). Stimulus generalization as a function of verbal reinforcement combinations. **Journal of Experimental Psychology, 48(6)**, 433-436.

Buss, A. H. & Buss, E. (1956a). The effect of verbal reinforcement on conceptual learning. **Journal of Experimental Psychology, 52(5)**, 283-287.

Buss, A. H., Braden, W., Orgel, A., & Buss, e. (1956b). Acquisition and extinction with different verbal reinforcement combinations. **Journal of Experimental Psychology, 52(5)**, 288-295.

Cohen, V. B. (1985). A reexamination of feedback in computer based instruction: Implications for instructional design. **Educational Technology**, 33-37.

Collins, M. T. (1984). The effectiveness of computer delivered correction procedures on low-performing secondary students' reasoning skills. **DAI, 45/12A**, Doctoral dissertation, University of Oregon.

Gaynor, P. (1981). The effect of feedback delay on retention of computer based mathematical material. **Journal of Computer Based Instruction, 8(2)**, 28-34.

Gilman, D. A. (1969). The effect of feedback on learners' certainity of response and attitude toward instruction in a computer assisted instruction program for teaching science concepts. **Journal of Research in Science Teaching**, 6 171-184.

Kowitz, G. T. & Smith, J. C. (1985). The dynamics of successful feedback. **Performance and Instruction Journal**, 4-6.

Kulha[illegible], R. W. & Anderson, R. C. (1972). Delay-retention effect with multiple-choice tests. **Journal of Educational Psychology, 63(5)**, 505-512.

Kulhavy, R. W. (1977). Feedback in written instruction. **Review of Educational Research, 47(1)**, 211-232.

Land, M. L. Effects of instructional feedback on student achievement with microcomputer lessons. Paper presented at the annual meeting of the Association for Educational Data Systems, New Orleans.

Lasoff, E. M. (1981). The effects of feedback in both computer assisted instruction and programmed instruction on achievement and attitude. **DAI, 42/04A**, Doctoral dissertation, University of Miami.

Lee, O. M. (1985). The effect of type of feedback on rule learning in computer based instruction. **DAI, 46/04A**, Doctoral dissertation, Florida State University.

Noonan, J. V. (1984). Feedback procedures in computer assisted instruction: Knowledge of results, knowledge of correct response, process explanations, and second attempts after errors. **DAI, 45/01A**, Doctoral dissertation, University of Illinois.

Rankin, R. J. & Trepper, T. (1978). Retention and delay of feedback in a computer assisted instructional task. **Journal of Experimental Education**, 67-70.

Robin, A. (1978). The timing of feedback in personalized instruction. **Journal of Personalized Instruction, 3(2)**, 81-88.

Roper, w. J. (1977). Feedback in computer assisted instruction. **Programmed Learning and Educational Technology, 14(1)**, 43-49.

Sales, G. C. (1985). The effect of learning style and type of feedback on the achievement of pre-service teachers in a computer based lesson. **DAI, 46/10A**, University of Minnesota.

Sfondillas, J. S. (1985). The effects of combining routine strategy and the corrective feedback paradigm in a computer based lesson. **DAI, 46/11A**, Doctoral dissertation, University of Illinois.

Sturges, P. T. (1972). Information delay and retention: Effect of information in feedback and tests. **Journal of Educational Psychology, 63(1)**, 32-43.

Surber, J. R. & Anderson, R. C. (1975). Delay-retention effect in natural classroom settings. **Journal of Educational Psychology, 67**, 170-173.

	Rote, knowledge, discrimination learning	Higher cognitive learning
Immediate vs delayed	Immediate: Kuhavy et al. (1972) Delayed: Bardwell (1981) *Gaynor (1981) [negative effect] *Roper (1977) No significant difference: *Arnett (1985) *Beaulieu (1985) *Gaynor (1981) *Sales (1985)	Delayed: *Gaynor (1981) Sturges (1972) *Rankin (1978) Robin (1978)
[Right-wrong, Right-none, Wrong-none]	Right-none: *Sales (1985) [NS] Wrong-none (error identification): Buss et al. (1954) Buss et al. (1956a) Buss et al. (1956b)	*Lee (1985) [all SS- NS between KR, KCR, & error identification; high ability SS- significant]
results (KR); error detection]	Error detection (informational): *Anderon (1985) *Arnett (1985) [NS; combined KR and informational *Bumgarner (1984) *Gilman (1969) *Roper (1977) [combined KR and informational] *Sfondillas (1985) KR vs informational: *Armour-Thomas (1984) [NS] *Bumgarner (1984) *Collins (1984) *Land (1986) [NS] *Noonan (1984)	

* = CAI
NS = not significant

Table 1
Feedback Research

	Informational Feedback		
knowledge of results [KR]	knowledge of correct results [KCR]	error correction	additional information and/or branching
<------------------	------------------	------------------	------------------>
<------------------	------------------	------------------	------------------>
"Incorrect."	"Answer (a) is correct."	"CH4 is a hydrocarbon because it contains both carbon and hydrogen."	"C2H6 is also a hydrocarbon."

Figure 1
(Arranged according to the degree of informativeness of the message NOT necessarily by effectiveness.)

PREPARATION FOR COLLEGE COMPUTER SCIENCE

Harriet G. Taylor, *Chair*
Louisiana State University
Baton Rouge, LA

ABSTRACT

During the last decade, the number of freshmen entering college with prior computing experience has been steadily increasing. Many universities now expect or require high school computer science coursework for college entrance, particularly for students pursuing technical degrees. Many of the factors that determine how the student performs in his college computer science classes may well be determined by the student's pre-college preparation for studying computer science.

The presenters will discuss many of the elements that could lead to successful completion of a college level computer science curriculum. Topics will include:

1. Exit skills needed by computer science graduates;

2. Comparison of high school and college computer science classes;

3. Ways that a high school teacher can help students prepare for college computer science courses;

4. Factors other than computer science classes that might have a bearing on performance in college computer science.

The session will conclude with a general dialog with the audience about the basic issues of pre-college versus college computer science and how educators at all levels can work together to further computer science as a discipline throughout the entire educational system.

PANELISTS

John D. Lawson
Chair, ACM Careers in Computing Project
Towson State University
Towson, MD

Cathleen Norris
Coordinator of the Undergraduate Program in Computer Science
North Texas State University
Denton, TX

Jean B. Rogers
Chair, ACM Elementary and Secondary Schools Committee
Stanford University
Stanford, CA

SPONSOR: SIGCSE

CD/ROM, CD-I, AND THE "INFORMATION REVOLUTION"

Francis A. Harvey
Leadership, Instruction, and Technology Department
College of Education
Lehigh University
Bethlehem, PA 18015

ABSTRACT

Recent advances in optical disc storage technology have potential for far-reaching and significant impact on education and learning, both in traditional formal settings and in other settings. This paper examines the potential influence of two forms of optical disc storage technology: CD/ROM (Compact Disc -- Read Only Memory) and CD-I (Compact Disc -- Interactive). The paper argues that CD/ROM and CD-I represent an early phase of an "Information Revolution" analogous to the "Computer Revolution" of the past decade.

Both CD/ROM and CD-I technologies are based on optical storage of digitally encoded information. They differ significantly, however, in the current state of their development, in the type of user for which they are being targeted, in the environments in which they are intended to be used, and in the types of applications which are being developed for them. CD/ROM is a currently existing technology intended for use as a computer peripheral for mass data storage and retrieval. CD-I is a proposed standard for a stand-alone system (although with a built-in microprocessor) designed primarily for home use, which will be available in late 1987. CD-I will be able to incorporate large amounts of high-quality sound, still video, and computer-generated and computer-manipulated graphics in an environment which is potentially very interactive.

A technical description of CD/ROM and CD-I is given, and the relative strengths and weaknesses of each are analyzed and compared in terms of the current state of development of each technology. A range of currently existing and projected examples of applications which imply fundamental changes in the uses of technology for education are presented. The potential impact of CD/ROM and CD-I on the content, form, management, setting, and delivery of instruction is discussed.

INTRODUCTION

The "Computer Revolution" of the 1970s and 1980s

The "Computer Revolution" which followed the introduction of personal computers in the middle 1970s brought about significant changes in all aspects of society, including education. For the first time powerful symbol manipulation capabilities were available to and under the control of the common person, teacher, or student, rather than being limited only to "professionals" or "experts."

The computer applications developed for business or personal productivity were designed for processing and manipulating data, either in the form of text (for word processing or similar programs) or numbers (electronic spreadsheets, data base management programs). Some graphics applications were also developed, for example, for the graphing of spreadsheet or data base data or "paintbox" programs for producing computer graphics.

In education, the majority of instructional applications developed were drill or tutorial programs oriented toward improving students' mathematical computational or language arts skills, or for expanding students' knowledge about science, social studies, or other subjects. Graphics were used principally to illustrate a point or to add interest or motivation to a text- or number-oriented presentation.

In summary, most of the power of the computer which brought about the changes which took place in schools, in the workplace, and at home as part of the "Computer Revolution" was devoted to number-crunching and "text-crunching" applications. The impact of the "Computer Revolution" on education can be seen by the fact that there are were over one million microcomputers in use in schools in 1985 (compared to 52,000 in 1980) and that over fifteen million K-12 students used microcomputers in schools last year. The impact on business and industry is reflected in the estimate that by 1990 over 80% of all jobs will require some knowledge of computers.

The "Information Revolution" of the 1980s and 1990s

Recent advances in optical disc storage technology have potential for far-reaching and significant impact on education and learning, both in traditional formal settings and in other settings. In this paper I will analyse and compare two forms of optical disc storage technology: CD/ROM (Compact Disc -- Read Only Memory) and CD-I (Compact Disc -- Interactive). I will argue that the widespread availability of CD/ROM and CD-I hardware and software could bring about an "Information Revolution" with an impact for greater than the impact of the "Computer revolution." The point of view presented is that CD/ROM and CD-I potentially represent not only new but radically different educational technologies. These technologies could allow the common person -- a teacher, a student, or an individual -- to access and manipulate directly, easily, and relatively cheaply vast amounts of information which were previously restricted to "experts" or "professionals," without using an "expert" as intermediary.

Four major changes which potentially will take place when CD/ROM and CD-I become widely used will contribute to the "Information Revolution." First, the amount of information to which a person has access will increase dramatically, approaching the amount of information now available only to experts. Second, the variety of information will expand greatly, to include not only traditional text and numerical data but visual and auditory information of all types: pictures, charts, diagrams, still and animated computer-generated graphics, still and motion videos, voice, music, and other sound information. Third, the search and retrieval capabilities of information systems will become more powerful, more efficient, and easier to use. Fourth, the level of interactivity between the user, the information system, and the information itself will be much higher, so that the user will be able to not only examine but also manipulate and modify the many types of information available.

An Overview of CD Technology: CD/ROM and CD-I

CD/ROM (Compact Disc -- Read Only Memory) and CD-I (Compact Disc --Interactive) are two particular examples of CD (compact disc) technology. CD is a generic term that refers to any technology based on optical storage of digitally encoded information that uses 12 centimeter (4.75 inch) diameter optical discs. CD-Audio is the most popular example of CD technology, and its popularity has helped lower the cost of CD players considerably.

Digital information is encoded onto a plastic CD platter by burning tiny pits into the surface of a master disc using a high-powered laser. Each pit is 0.12 microns (one micron is one-millionth of a meter) deep and 0.6 microns wide. One 4.75 inch disc can store 550 million bytes of digitally coded information.

Copies of the master disc are produced using technology similar to that used to produce phonograph records. The cost of mastering an optical digital disc is now about five thousand dollars. The cost of producing copies in quantity can be less than a dollar each.

Compact disc players use a low-power laser beam to scan the surface of the spinning disc. A system of lenses and mirrors directs the light reflected from the pits and "lands" between the pits on the disc into a series of light-sensitive diodes. Signal processing circuitry converts the pits and lands to binary 1's or 0's and sends these bits to a computer or other device. Since the surface of the disc is never touched by anything but the laser beam, optical discs could in theory last forever.

CD/ROM and CD-I technologies differ significantly in the current state of their development, in the type of user for which they are being targeted, in the environments in which they are intended to be used, and in the types of applications which are being developed for them. CD/ROM is a currently existing technology intended for use as a computer peripheral for mass data storage and retrieval. CD-I is a proposed standard for a stand-alone system (although with a built-in microprocessor) announced in 1986 by Philips and SONY. CD-I is designed primarily for home use for both educational and entertainment applications. The first CD-I systems are expected to be available in late 1987. CD-I will be able to incorporate large amounts of high-quality sound, still video, and computer-generated and computer-manipulated graphics in an environment which is potentially very interactive.

In this paper a technical description of both CD/ROM and CD-I will be given, and a range of currently existing and projected applications will be discussed. The relative strengths and weaknesses of CD/ROM and CD-I will be compared in terms of the current state of development of each technology. The implications of the introduction of and widespread use of CD/ROM and CD-I for education and for educators will then be discussed.

CD/ROM TECHNOLOGY AND APPLICATIONS

Overview

Technical description. CD/ROM discs are laser optical discs designed for use as peripheral devices for computers. They are used for storage of large amounts of digital data, over half a billion bytes per disc. One CD/ROM disc holds the equivalent of 1,500 MS-DOS microcomputer floppy disks (or over 3,000 Apple IIe disks). It would take one secretary typing ninety words per minute, eight hours a day more than eight years to type the amount of information which could be stored on one CD/ROM disc.

CD/ROM discs are read using a player connected to a microcomputer. The player, depending on the model and type of microcomputer, can be a stand-alone peripheral device connected to the computer's serial port or a built-in device designed to fit inside a microcomputer in place of a floppy disk drive. Some CD/ROM players are designed to use the high-speed SCSI (Small Computer System Interface) standard interface found on Apple Macintosh Plus, Apple IIGS, and other newer computers.

The CD/ROM market is expanding rapidly; less than 3,000 CD/ROM players were sold in 1985, but it is projected that over 350,000 will be sold in 1988. Currently less than two percent of microcomputer storage uses CD/ROM; by 1990 it is expected that this percentage will increase tenfold.

Advantages of CD/ROM

The major advantage of CD/ROM its large capacity storage. Much more information can be provided to users in a much smaller space. This is an important factor for libraries, for example, which must cope with ever-growing collections within a finite (usually fixed) space.

A second major advantage of CD/ROM is the permanent nature of its medium. Information, once put on the disc, can not be erased or changed, either accidentally by a naive user or intentionally by a knowledgeable (or malicious) user. For many applications, such as reference bases which are shared by several users or for archival information which does not change, this is desirable.

Since CD/ROM systems are microcomputer-based, they also provide a high degree of user control and accessibility. The information on a CD/ROM is available to the user at any time, without having to depend on phone lines, account numbers, or passwords for access to mainframe data base systems. Retrieval of information is much faster than the retrieval of the same amount and type of information by hand.

Other advantages of CD/ROM include their relatively low cost (being based on CD-Audio technology), durability, and convenience. They are also flexible in terms of the variety of media which can be stored on them.

Limitations of CD/ROM

One advantage of CD/ROM, that the data can not be changed, can in some applications be a disadvantage. Updating a CD/ROM disc involves re-mastering the entire disc with the old as well as new information. CD/ROM may not be the most appropriate format for data which changes rapidly, or for data to which users are constantly adding.

Access time for CD/ROM discs is slow compared to magnetic (floppy or Winchester) disk systems: typically it takes about one second to locate a file on a CD/ROM disc and position the laser beam for reading it.

Applications of CD/ROM in Business and Science

CD/ROM is appropriate for any application in which there is a need for selective referencing of a large amount of relatively stable information. For example, DATEXT, Inc. has produced a corporate information data base for use by financial analysts. The DATEXT disc holds information on over 10,000 companies in 900 lines of business in 50 industries. Corporate Technology Information Services has produced a similar data base on companies involved in high technology. Lotus Development Corporation began distributing financial information and other information on stocks and bonds on CD/ROM disc in November, 1986.

CD/ROM can be a durable, compact and inexpensive software distribution medium. Reference Technologies, for example, distributes the CD/ROM DataPlate, which holds 8,800 public domain software programs for MS-DOS computers.

CD/ROM discs can store maps, diagrams, and other graphical information as well as numerical data and text. The United States Geological Survey (USGS) has recently produced a CD/ROM disc for internal use which holds geological information on several large land areas as well as bibliographies, all of the policies and procedures of the USGS, LANDSAT data for several areas, the USGS Cartographic Catalog, and much other information. The disc also includes a "software sampler" of seven different commercial full-text retrieval software packages.

Applications of CD/ROM in Education

CD/ROM has yet to reach its full potential in education, due to the relatively high cost and low utilization rate among schools. The cost will drop quickly, however, as the number of educators using CD/ROM players increases, and the number of applications will increase.

The major educational application of CD/ROM at present in the Grolier Electronic Encyclopedia. One CD/ROM disc holds the entire text of the 20-volume Academic American Encyclopedia, over nine million words in more than 30,000 entries. This is the equivalent of 10,000 pages, or over two feet of shelf space of regular books. A specially designed index pinpoints the location of every occurrence of every word in the encyclopedia. The data and the specially designed index together use only one-fifth of the total space available on the CD/ROM disc.

The Grolier Electronic Encyclopedia can be used with an IBM-compatible microcomputer or an Apple (with a special adapter card) and any standard CD/ROM player. The Knowledge Retrieval Software (KRS) provided, developed by Activenture, allows the user to search the encyclopedia by article title, by article text, or by bibliographies and tables. The user can ask to select all articles which contain a specific word, combination of words or phrases, or even partial words.

Boolean search capabilities are also included. For example, the user can select all articles which include the word buildings or the word structures. The user can ask for all articles which include both the word birds and the word prehistoric. The user can also negate terms, for example, by specifying occurrences of the word flowers but not the word roses.

A "browse" capability allows the user to examine the text of articles selected. If desired, the text can be printed on a printer, or even written to a floppy disk file for later incorporation into a word processor document.

The ERIC data bases (Resources in Education (RIE) for education reports, and Current Index to Journals in Education (CIJE) for journals) are now available on CD/ROM.

TESCOR markets the First National Item Bank and Test Development System, a CD/ROM disc containing tens of thousands of validated test items from all curriculum areas. The test items are linked to thousands of curriculum-related objectives. Teachers may search for objectives which match a particular curriculum area, age level, and other characteristics, then select from a group of test items which match the objectives. The items can be combined into tests and printed with the same computer.

Ibycus Systems has produced the Scholarly Personal Computer for use by scholars working with ancient Greek, Latin, Hebrew, and Coptic literature. It includes a CD/ROM disc with the works of Homer, Sophocles, Plutarch, Plato, Aristotle, Virgil, and other Coptic and Latin authors. The New Testament in Greek and the Old Testament in Greek and Hebrew is also included.

Biolibe is a recently announced biological encyclopedia on CD/ROM, which will hold over 6,000 images of insects, flowers, birds, animals, and plants. The disc will also include text information integrated with the images as well as graphics and a data base management program which will be able to answer questions such as "What are the red flowers that bloom in Illinois?" or "What are the big grey animals?" as well as more technical questions phrased in terms of genus, species, or phyla.

GEOdisc, to be developed by GEOVision, Inc., will contain selected maps and geographical data for the continental United States.

CD/ROM discs can also be used as support devices for laser optical discs (analog videodiscs). One CD/ROM disc can store more than thirty-five hours of compressed speech, which could be used to provide narration or other information to accompany still or motion video segments from the videodisc. CD/ROM discs can also be used to provide reference base support for videodiscs. Such integrated systems, however, are usually expensive. The Electronic Delivery System, for example, offered by Online Computer Systems, includes an MS-DOS computer, videodisc player, CD/ROM player, light pen, and software. It is designed for corporate and industrial training and costs seven thousand dollars.

Summary of CD/ROM Applications

CD/ROM is clearly an idea which has arrived. Many applications are available, and many more have been announced. To data, most of these applications are in business or corporate environments, but the number of applications for educational use is growing rapidly, and will continue to grow as educators recognize its power. CD/ROM is appropriate for any situation in which there is a large amount of fairly static information to which users need random access. CD/ROM can make this access easier and more efficient, and can extend the accessing capabilities of the user through more powerful search and retrieval software. As such, the time and energy of the user can be devoted to analysis and synthesis of the information available. This will imply the need for a shift in emphasis on skills taught from lower-level tasks such as "how to find an article on ..." to higher-level tasks such as "how to decide which of 45 articles on ... is relevant, and how to integrate the information from those relevant articles into a cohesive summary."

CD-I TECHNOLOGY AND APPLICATIONS

Overview

Technical description. CD-I is a proposed standard for storing text, graphics, and sound information on CD discs announced as a joint venture of SONY and Phillips in 1986. A CD-I system will be a mass market product which will incorporate a compact disc player, a Motorola 68000 microprocessor, and a mouse or joystick input device. It will be designed to connect to a home TV set and stereo system. A computer is not needed, although with the addition of a keyboard and appropriate software the CD-I system could incorporate the capabilities of a personal computer.

CD-I includes standards for audio, video still pictures, animation, and text. A single worldwide standard will be followed, so all CD-I discs will play in all CD-I players. It is possible that CD-I players will also be able to play CD-Audio discs.

CD-I will provide four audio formats, from high-quality stereo to "speech mode" with quality equivalent to AM radio. Ten hours of stereo speech, twenty hours of monophonic speech, or 74 minutes of high quality stereo music can be stored on one disc. Sound formats can be mixed and matched on the same disc.

CD-I video formats include natural video pictures, user-manipulable graphics, and quality graphics. Natural pictures will resemble high-quality video stills. User-manipulable graphics will be lower in quality, but will still provide very high resolution and a large number of colors. Quality graphics are more "cartoon-like"; however, they offer the capability of displaying full-screen animation, which could be combined with various sound tracks.

CD-I text will resemble high quality text as seen on broadcast television rather than "computer screen text." Up to 600 million text characters can be stored on one disc. With the appropriate software on the disc, the user can search the text and retrieve and manipulate parts of it.

The built-in Motorola 68000 microprocessor will use an OS-9 based operating system called Compact Disc Real Time Operating System (CD-RTOS). The operating system will be complex, and will require considerable RAM memory in order to function effectively.

Limitations of CD-I

CD-I uses digital, not analog, encoding of the information on the disc. Due to the speed of transferring such information from the disc and converting digital information to video, it will not be possible to produce true motion video with CD-I. Many instructional developers consider this a major drawback; others, however, feel that the other features of CD-I will allow the instructional developer to overcome this limitation.

The major limitation of CD-I is that it does not yet exist. A standard has been proposed, and several very large corporations have made a commitment to developing it. On paper, the CD-I system is impressive. However, it is impossible to predict the compromises which may become necessary in developing CD-I and therefore the actual capabilities of the CD-I systems actually delivered.

CD-I offers the promise of the ability to deliver vast amounts of informational or instructional material in many different modes. It further offers a sophisticated computer operating system to manipulate that information and to handle complex interactions with the user. Potentially, the user has more power to manipulate and modify the materials being presented than in any other technological system, and thus, in principle, learing with CD-I can be truly user-directed, active, and interactive learning. Educators have reason to hope that the final version of CD-I approaches the lofty goals which its developers have set for it.

Applications of CD-I

The CD-I system will be designed to produce interactive programs for education, entertainment, and reference. The CD-I systems could be used to play extremely high-fidelity music while simultaneously viewing the music score on the TV screen, or video stills, graphics, or other visuals about the composer or the period of history in which the music was composed.

CD-I will allow the development of interactive computer/video games which use TV-like images, and which talk to the user as the game progresses.

A CD-I disc on first aid could be consulted in an emergency. The same disc could include tutorial software for instruction in a non-emergency situation.

A CD-I disc could be a "talking book" with twenty hours of speech. If desired the text of the speech could be displayed simultaneously.

A CD-I disc could be the "coffee table book" of the future. One CD-I disc could include thousands of high-quality video stills. The user could view the video pictures, switch to related text and read it, or listen to spoken information or related music or other sound while viewing the video still images. The user could display a particular image (e.g., the Mona Lisa), modify it using an interactive video paintbrush, and save the modified image (on a computer disc used as a peripheral to the CD-I unit).

Educational Applications of CD-I

It is difficult to discuss applications of a system which does not yet exist. The descriptions of the system capabilities given above provide an indication of the educational capabilities of CD-I. The following are examples of CD-I applications which are within the capabilities of CD-I technology:

The Time Machine. A time-based history of the world, linked to an everchanging geological/geopolitical map. The map could change at the rate of one year per second, beginning in 500 BC. The user could stop and view any year, and examine either "headlines" describing the major events of that year or listen to audio segments which provide additional detail about the year's events.

Genealogical Relationships of World Leaders. A description of the family trees of current world leaders, going back as far as existing records permit. The program could be used as a reference work or played as a game. The working title of this program is "Princess Di is Related To Chiang Kai-Shek."

Design-A-World. The user would specify the characteristics of a closed environment, the types of living organisms in the environment, and the characteristics of each living organism in the environment. Alternately, the program would present "standard" sets of characteristics which the user could modify. The program would then put the student in the environment and allow him or her to "live" there for a period of time. The user could observe the rise and fall of various populations. The user could also switch points of view, for example, to "be a hawk" for a period of time, to see the world and its changes from a different perspective.

The above are offered as examples only, and merely scratch the surface of the potential capabilities of CD-I. Much more is possible, for example, in foreign language instruction of in music instruction.

Advantages of CD-I for Education

As was the case to some extent with microcomputer software, the development of a substantial home market for CD-I educational software could increase the number and quality of software applications which could be used in schools. Bank Street Writer and Rocky's Boots are examples of this type of migration of quality microcomputer software.

It is also possible that "entertainment" applications for home use will become more "educational." Computer games, activity games, and board games could all become more interactive. Interactive dramas could be developed in which the user chooses the plot and other dramatic elements, then interacts with the other characters in the play presented.

Software distribution on CD-I discs could reduce or eliminate the problem of use of illegal copies of software, since the disc itself would be required in order to use the software. The development of a mass market would mean reduced prices per unit for quality software.

A COMPARISON OF CD/ROM AND CD-I

CD/ROM is conceived and marketed as a computer peripheral. CD-I is designed and will be marketed as a stand-alone consumer product, similar in concept to a stereo system or videocassette recorder. Although computer-like peripherals (keyboards, magnetic (floppy or hard disk) dirves, printers, etc.) will no doubt become available for it, the standard configuration will not include computer capabilities.

CD/ROM has been on the market for several years, and many proven products are available in CD/ROM format. CD-I will not be available until late 1987, and large numbers of systems will probably not be available until a year or two after that. Software developers who closely watch "the bottom line" may be deterred from entering a market which will take some time to develop.

CD/ROM can store any type of digital information; CD-I stores audio, video, text, and software in predefined formats. CD/ROM is aimed primarily at the business market, while CD-I is aimed at homes, schools, and libraries.

In summary, CD/ROM and CD-I are two different products aimed at separate and distinct markets, although there is considerable overlap in those markets, especially in the educational area. CD/ROM and CD-I will coexist and support each other, rather than competing with each other. In the best of all possible worlds, a CD/ROM disc would be compatible with CD-I players, so the same player could be used for CD-Audio, CD-I, and CD/ROM applications.

CONCLUSIONS

In this paper I have presented an overview of two forms of optical disc storage technology, examples of existing and potential applications of each, and the present and future impact on education and the educational process. The thesis of this paper is that changes in the content, format, mode of delivery, setting, mangagement, and control of education which will be possible with CD/ROM and CD-I will, if realized to any significant degree, result in fundamental and radical changes in education, changes that will be revolutionary rather than evolutionary in nature. We are now in the very beginning phases of these changes, but we can expect the changes to be comprehensive, extremely rapid, and inevitable.

As educators we now have a "window of opportunity" for directing and influencing the changes that will take place. In order to do so, we must recognize that the changes are coming, educate ourselves about the capabilities of the new technologies, and be forthright in assuming responsibility and leadership for the reshaping of education which could take place in the very near future.

CHILDREN'S USE OF A CD-ROM ENCYCLOPEDIA: A LOOK AT ELECTRONIC INFORMATION-SEEKING STRATEGY

Gary Marchionini
4121 Hornbake
University of Maryland
College Park, MD 20742

Diane Patrick
Educational Technology Ctr.
Benjamin Building
University of Maryland
College Park, MD 20742

Jerry Teague
3203 Benjamin Bldg.
University of Maryland
College Park, MD 20742

Abstract

Electronic information-seeking skills are increasingly important for today's students. A study of how elementary school students used a full-text, electronic encyclopedia was conducted. Two groups of students (third/fourth-graders and sixth-graders) were compared on level of success in finding information for two tasks, time taken to conduct the searches, and number of search moves made. Two tasks, open-ended and closed, were compared on the three criterion variables. Older students were more successful than younger students, but two-thirds of all searches were successful. No differences between the tasks with respect to success were found, but the open task took significantly longer and more moves to conduct. Teachers are encouraged to consider skills for filtering and evaluating information in addition those for simply recalling information.

Skills for retrieving, filtering, and organizing information are critically important in an information driven culture. While much of a learner's early education has always been devoted to techniques for asking questions, using basic reference tools, and ordering collections of data, two trends are beginning to alter the basic methods we employ to seek and use information. Consequently, these trends will have a significant impact on the skills for accessing information that we teach our students.

First, the total amount of information produced is growing exponentially. This phenomenon produces both positive and negative effects. On the one hand, finding the required information will be more complex, since there is more to examine and filter out. Presently, the emphasis is often on recall—maximizing the amount of relevant information; however, in an information rich environment, there is greater need for precision-minimizing the amount of irrelevant information retrieved. Techniques for filtering information and avoiding information overload must be developed and taught in school if students are to be prepared to live and work in an information society.

Second, the development and proliferation of personal computers, telecommunications networks, and mass storage devices like optical disks have allowed users to store, retrieve, and manipulate huge quantities of information quickly and easily. This situation also brings problems as well as benefits: although the ability to access more information and manipulate it electronically promises comprehensive, well-organized work, instruction in the procedures required to access and manipulate the information electronically must be added to the already cluttered curriculum. These procedures go beyond the simple mechanics of using hardware to more complex and less understood techniques for operating with electronic rather than print or oral forms of information.

The purpose of this paper is to report the results of research that explored how elementary school students used an electronic encyclopedia. The effects of type of search task and grade level on success, time taken to complete the search, and number of search functions used, were examined and patterns of information-seeking and system use were observed. The results have implications for curriculum specialists, systems designers, and designers of instructional programs.

Background

Three areas of research and development contributed to the definition and execution of this research: existing curricula for information-seeking skills; identified strategies for information-seeking in electronic environments, especially online searching; and such technical and conceptual developments as full-text retrieval systems, and optical disk technology. Results from these areas provided a framework for building a cognitive theory of information-seeking and posing the immediate questions of how to deal with the information explosion and emerging technologies in schools.

Information-seeking skills

Information-seeking skills are basic to the thinking process and thus are taught as part of most subject areas in the early grades. Some research in information-seeking during reading has been conducted. Guthrie and Mosenthal (in press) report that locating information is the primary reason adults read; point out that information gathering skills are assumed in many curriculum guides but that little teaching guidance is provided; and present a framework for literacy that includes locating information as a distinct, major factor. They point out the need for considering the methods of locating information as a distinct process in overall reading comprehension.

The great attention to database use at all grade levels today is testimony to the recognition that electronic information-seeking skills are deemed important. Suggestions for using database management software to promote thinking skills have been provided by Parker (1986), Schiffman (1986), and Cutts and Garvin (1986); but no evidence of effects or particular objectives for optimizing database searching are provided.

Library media specialists have long been involved in teaching information-seeking skills. Although skills for using new technologies are included in today's scope and sequence charts (e.g., Walker & Montgomery, 1983), there has been little experience with use of telecommunications or CD-ROM databases in schools. In addition, most studies have focused on print or oral information formats. Thus, there is no concrete experience base to provide guidance for teaching electronic information-seeking skills. Mancall, Aaron and Walker (1986) suggest that most instructional programs for information-seeking are library-centered rather than information-centered and provide an excellent framework for considering information-seeking skills as part of the overall problem of teaching thinking skills. There is a clear need for both experiential and theoretical work in learning and teaching information-seeking skills in electronic environments.

Online searching

There have been many studies of information specialists' use of electronic information sources. These specialists use bibliographic, numeric, and text databases to retrieve both secondary (citations, bibliographies, etc.) and primary (facts, cases, etc.) references. Since most of the more than 3000 online databases (*Directory of Online Databases*, 1986) are bibliographic, most studies have examined librarians' strategies for searching bibliographic databases. For example, Markey and Cochrane (1981) have identified five overall strategies that specialists use for searching; Bates (1979) proposed 29 search tactics that help professional searchers locate information; and Fidel (1982) has identified two styles of information-seeking. Although these works provide an excellent beginning, much remains to be learned about how expert searchers seek information in electronic databases. Moreover, there are few studies involving full-text databases and none with CD-ROM databases and microcomputer-based search systems. Until studies with novices in school settings are conducted, teachers and librarians designing lessons and materials for students must be guided by the tactics, strategies, and styles identified for expert users rather than by knowledge related to students' abilities and needs.

Fidel and Soergel (1983) have proposed a framework for studying online searching that was adapted for the research reported here. The framework as adapted for novice users predicts that the outcome of an information search is dependent on the interaction of five factors: the searcher, the search problem (task), the setting, the database (source), and the search system. For this work, setting, database, and search system were controlled, since all subjects used the same database and search system in the same setting.

Full-text electronic systems

Full-text databases are databases that contain entire documents. Full-text databases are intuitively appealing because they are directly analogous to print forms. The computer can be directed to "read" the entire text rapidly and "look" for well-defined strings of characters, for example, words or phrases. As users of full-text systems have discovered, however, specifying well-defined strings for search topics is easier said than done. Tenopir (1984) points out that problems remain for both users and designers of full-text systems.

The development of CD-ROM for text has caused widespread interest in the publishing industry because tremendous amounts of text can be cheaply delivered in electronic form. A single CD-ROM disk can store about 600 megabytes (about 200,000 pages of text). Costs for disks are only $4000 for mastering and $6.00 a copy in quantities as small as 5000 (Optical Information Systems, 1986). As prices for players continue to fall and more publishers release text works on disk, most school libraries will certainly acquire such systems.

The first CD-ROM disk aimed at the popular market was Grolier's *The Electronic Encyclopedia*. The twenty-volumes of text (no pictures) for the entire encyclopedia takes only about 60 megabytes of the disk. A huge index (including all the words in the encyclopedia indexed to every occurrence in the entire work) requires about 50 megabytes and is included on the CD-ROM along with the articles. A computer program that comes with the disk enables the user to search this index to find all occurrences of a word, phrase, or combination of words or phrases in the entire encyclopedia. The program thus allows the user to search the encyclopedia in ways not possible using print encyclopedias and manual search techniques. Moreover, the database and software cost about one-third as much as the print version of the encyclopedia. As other publishers make materials available in similar fashion, the implications for information-seeking both in and out of school are obvious.

The effects of new technologies and techniques for seeking information on teaching students about information-seeking will be determined over the course of the next decade as computers become more integrated into the overall school program. A first look at some of the factors involved motivated this study while the approaches taken and results produced by research in online searching guided its execution.

Method

Two groups of elementary school students in a suburban school system worked in pairs to conduct two assigned searches in the electronic encyclopedia. The study took place during the spring of 1986 over a period of two months. Our previous work (Marchionini & Teague, in press) demonstrated that students as young as second and third grades could successfully search for information in this encyclopedia through an electronic information utility. In this study we controlled the search tasks by designing two distinctive search problems and assigning them to students. All searches were observed by a researcher and all keystrokes captured unobtrusively by the computer. Students recorded the information they found as they conducted their searches. Two forty-five minute large-group sessions were conducted for each of the two groups to introduce them to the use of the electronic encyclopedia system. The major factors considered are detailed below.

Searcher (user)

A combination class of third and fourth-graders and a class of sixth-graders participated in the study. The students were enrolled in a talented and gifted program in the school and all had had at least keyboarding experience with microcomputers. Teachers were asked to pair the students in the respective classes to maximize cooperation and compatibility. A total of 26 pairs (14 in the third/fourth grades and 12 in the sixth grade) were formed and each pair came to the media center twice to conduct its two searches.

Task

Two search problems were stated in paragraph form and given to the subjects when they arrived. One task was designed to be closed in that it required subjects to find a fact (the year speed skating was introduced into the Olympics). The second task was designed to be open in that it required subjects to find information that was not strictly delimited in quantity (information about women who have traveled in space). The tasks were pilot tested with a variety of subjects to assure both that multiple search paths were possible and that the problem statement was clear.

Database and search system

Database and search system factors were controlled in that all subjects used the *The Electronic Encyclopedia* and accompanying *Knowledge Retrieval System*. Although the system supports access to the encyclopedia through article title only ("browse" mode), we did not demonstrate nor did subjects use this mode. Rather, we focused on the full-text mode termed "Word Search". The search system is very powerful--providing boolean connectives (and, or, not), truncation and masking, proximity limitations, and scope limitations. The user controls all these features through ten function keys which at times take on different meanings: for example, at one point "F4" shows titles, while at another it displays the next page. The screen is quite "busy", providing six windows for the various options and feedback.

Outcome Measures

A simple index for success was determined for each search by counting a search successful if the correct fact (closed task) or a correct name (open search) was recorded by the subjects. Keystroke data were used to create an index of moves for each search by counting the total number of functions (not keystrokes) selected during a search. The complete set of moves considered was: original term, new term, broaden term, narrow term, synonym, reorder terms, change relation, show titles, and show text. These three measures--success, time, and number of moves were used to compare task and group. Relations among success, time, moves, task, and group were also examined.

Results

Group

Students were surprisingly successful in dealing with the complexity of the search system. This was probably due at least partly to the students ignoring complex features like scoping and proximity, and using only AND as a logical connective. One feature--changing the proximity limits for terms--was used by 12 pairs, seven sixth-grade pairs and five third/fourth grade ones. All these sixth-grade searches were subsequently successful and all the third-fourth grade searches were unsuccessful.

Two thirds (34) of the 51 searches completed were judged to be successful. Table 1 provides frequency counts for success by group.

Frequency of Success by Group

	Successful	Unsuccessful
Grades 3/4	12	15
Grade 6	22	2
Total	34	17

Note: *n*=51

The sixth graders were clearly more successful in finding information than the younger group. One third grade pair did not conduct their closed search due to absences. Because the success scale used was dichotomous, a Chi square value for the frequency data was computed to compare the groups. The value 10.7 (after Yates correction) was statistically significant at the .01 level. That the older group performed better than the younger group is not surprising, considering their additional experience with reading, information-seeking, and with the task domains included in the two search problems. Future efforts are needed to partition the effects of information-seeking experience and task domain knowledge.

The distributions of the two groups were compared on time and number of moves. The Kolmogorov-Smirnov Z value of 1.42 (p .04) was not statistically significant at the .01 level. It should be noted that 18 of the younger group searches were stopped. It is likely that greater differences would have been found had there been no time limits on searches. That younger subjects would take more time is predictable since their reading rates when examining articles were generally slower.

The number of moves taken to complete each search is directly related to the information-seeking process itself and thus is of interest as a dependent variable on which to compare groups. The mean number of moves for all searches was 21.0 (22.4 for the younger group and 19.5 for the older group). The Kolmogorov-Smirnov Z value was .64 (p .64), indicating no statistically significant differences between the groups. This is an interesting result because there is a strong negative correlation (-.43, p .01) between success and number of moves but not between group and number of moves (-.13, p .18). That is, a smaller number of moves is strongly related to success but weakly to age. Number of moves is thus an important factor to examine more closely in future studies of information-seeking strategies.

Task

Since the search problem has an important role in determining what information-seeking strategy is applied, the effects of task on success, time, and number of moves were also examined. Table 2 presents frequency data for success by task.

Frequency of Success by Task

	Successful	Unsuccessful
Open	17	9
Closed	17	8
Total	34	17

Note: n=51

The Chi square value of 0.0 (after Yates correction) indicates no differences between the tasks with respect to success. This is not surprising, since both questions were judged on a single "correct" response to each search.

Task did affect the time taken to complete a search. The mean time for the open task was 41.1 min. and the mean time for the closed was 30.7 min. The Kolmogorov-Smirnov Z value (1.60) was statistically significant at the .01 level. Since subjects were looking for multiple facts in the open task, it is not surprising that they took longer to search for those facts. This finding should also be reinforced by number-of-moves data, and this was the case. The mean number of moves for the open task was 24.7 and the mean number of moves for the closed was 17.2. The Kolmogorov-Smirnov Z value was also 1.60, statistically significant at the .01 level.

Conclusions and Recommendations

Overall, elementary school children were able to use a powerful electronic encyclopedia with success. No significant differences were found between tasks that were open-ended or closed. It appears that advanced search features can be used and presumably taught to sixth graders but that younger children may have difficulty applying such features. System designers developing software for elementary school students are advised to consider levels of difficulty in developing screen displays and default conditions. Teachers and curriculum specialists are encouraged to develop instructional activities which focus on the process of filtering and evaluating information rather than the mechanics of using a particular system. These results must be considered in light of the following limitations.

First, the students who participated in the study were of above average ability and thus generalization of the results should be done cautiously and only to similar groups. Secondly, the reading level of the encyclopedia text is aimed at junior high readers. Although the sixth graders had little difficulty reading articles, some of the third graders exhibited difficulty in reading the text.

Although these results cannot be generalized to all elementary school settings, they represent a first look at the effects of electronic information-seeking strategy. More detailed analyses of the information-seeking patterns exhibited by novice users are currently being pursued. The findings reported here, however, provide baseline data for teachers who wish to develop instructional materials for information-seeking in electronic environments and to designers of the electronic information systems that will enter the schools in increasing numbers. As teachers and students begin to take advantage of the unique capabilities of this technology to retrieve information, they should keep in mind both its potential and its pitfalls.

References

Bates, M. (1979). Information search tactics. Journal of the American Society for Information Science. 30 (4). pp. 205-214.

Cuadra/Elsevier (1986). Directory of Online Databases 7 (3).

Cutts, P., & Garvin, K. (1986). Teaching thinking with a data base. In W. Ryan (Ed.) Proceedings: National Educational Computing Conference 86 (p. 280). Eugene, OR: ICCE.

Fidel, R. (1982). Constructing a pattern model of man-machine interaction in online literature searching. (Doctoral dissertation, University of Maryland, 1982).

Fidel, R. & Soergel, D. (1983). Factors affecting online bibliographic retrieval: a conceptual framework for research. Journal of the American Society for Information Science. 34 (3). p. 163-180.

Guthrie, J. & Mosenthal, P. (in press). Literacy as multidimensional: locating information in written documents. Educational Psychologist.

Marchionini, G. & Teague, J. (in press). Elementary students' use of electronic information services: an exploratory study. AEDS Journal.

Mancall, J., Aaron, S. & Walker, S. (1986). Educating students to think: the role of the school library media program. School Library Media Quarterly, 15 (1). p. 18-27.

Markey, K. & Cochrane, P. (1981). Online training and practice manual for ERIC data base searchers. Washington, D.C.: NIE. (ERIC Document No. ED 212 296).

News and Notes—CD-ROM and Compact Disc. (1986) Optical Information Systems. 6 (4). p. 291.

Parker, J. (1986). Tools for thought. The Computing Teacher, 13 (8). p. 27-31.

Siegel, S. (1956). Nonparametric statistics for the behavioral sciences. New York: McGraw-Hill.

Tenopir, C. (1984). Full-text databases. In M. Williams (Ed.), Annual Review of Information Science and Technology. (pp. 215-246). Washington, D.C.: ASIS.

Walker, H. & Montgomery, P. (1983). Teaching library media skills. Littketon, CO: Libraries Unlimited.

PLANNING FOR COMPUTER ENHANCED INSTRUCTION

Gertrude W. Abramson
Baruch College
City University of New York
17 Lexington Avenue Box 505
New York, New York 10010

Abstract

Some of the educational software packages developed recently have been evaluated and found to be excellent resource materials for enhancing the learning and mastery of various components of the subjects taught in grades K-12. Computer educators frequently receive catalogs offering these packages that promise to revitalize learning in their class-rooms. However, specific suggestions for methods of integrating computing experiences into the curriculum rather than using them as "add-ons" have not been proferred.

Contemporary teachers require planning models to help them incorporate educational technology into the learning process. This paper presents a carefully guided working method for teachers to create lesson plans that integrate computer software use into daily instructional activities at both primary and secondary educational levels. The model begins with pre-instructional goal setting redefined as behavioral objectives tailored to specific student characteristics. Instructional planning includes a list of readiness and follow-up activiites and materials. Finally, post-instructional outcome assessments are written. Two examples of the model that were developed and implemented by classroom teachers are presented.

Introduction

Computers have revolutionized society reaching into all areas and eliciting tremendous publicity and attention; parents often recognize the potential of computers and lobby for computers in schools (Bork,1985; O'Shea & Self, 1983). This paper presents a working method of creating lesson plans that integrate computer software use into daily instructional activities and offers two examples of such plans that were developed and implemented by classroom teachers.

Software as a Teaching Tool

There are 10,000 or more educational software disks on the market today that range in technical and educational quality from poor to excellent. Bork (1985) expressed concern that the large number of inadequate pieces of educational software currently flooding the market will discourage users. He emphasized the need to distinguish between the potential for educational computing as an instructional delivery system for the future and today's reality. Accompanying the deluge of educational software are periodicals written for the classroom teacher such as Electronic Learning, Electronic Education and Classroom Computer Learning that discuss software teachers have found useful in their classrooms. The novice, who is overwhelmed by the plethora of offerings and catalogs, might begin integrating software in his classroom using work that has been recommended by others, assuming he makes the selection.

We should not expect teachers to produce their own software, the creation of which is costly and time-consuming and requires proficiency as a programmer and as an instructional designer. Neither should we assume that teachers who are often overburdened with non-instructional and clerical components of their jobs, will find the time to evaluate software. Furthermore, the purchase decision is frequently an administrative one that is made without including the teachers who are expected to use the materials in their classrooms. My students, all classroom teachers enrolled in Baruch College Graduate School of Education, were very surprised to discover that a comprehensive evaluation of software at any level can take from 20 to 40 hours of concentrated effort. Fortunately, educators involved with educational software have available extensive review literature from Educational Product Information Exchange (EPIE), Northwest Regional Education Laboratory's MicroSIFT Clearing House (MicroSIFT) National Council of Teachers of Mathematics (NCTM) Evaluation Checklist and Documentation Sheet and California Library Media Consortium for Classroom Evaluation of Microcomputer Courseware, to name just a few (Lathrop & Goodson, 1983).

It is ironic that the essential issue of student achievement is precisely the one that is ignored by the major compilers of educational courseware reviews. Bonner (1985), in discussing learning outcomes of computer-assisted instruction (CAI), quotes Sivin of EPIE, Weaver of MicroSIFT and Bitter of Arizona U. who agree that their evaluations are not designed to measure the effectiveness of a program in terms of whether or not student learning is taking place. They fall back on the time-worn argument of the difficulty of setting up control groups, pre- and post-testing and delineating and measuring the "higher cognitive" skills being developed with the CAI and stress instead the positive affective outcomes of the CAI experience. To successfully harness the power

3RS

inherent in our current educational technology, it is not sufficient to convince teachers that computers are easy to use or that their adoption will enrich the study of traditional subjects, it is crucial that teachers be helped to acquire the necessary skills and instructional support materials needed to enable the computer to be used as an aid to teaching and learning. Training in interpreting software reviews or in computer operation and disk manipulation are easily mastered first steps in preparing for the new educational delivery system.

Factors to Consider When Using Software

Having spent many years teaching programming languages to adults, I could not help but be impressed by how much easier and more successful teaching and learning became once structured programming was developed and established as the norm. I therefore determined to develop a procedural structure for the creation of computer-integrated lesson plans, described below, that incorporate educational technology in the learning process. Field testing of the components of the model with two semesters of Baruch College graduate students revealed several factors that must be considered when integrating software into the curriculum.

First, courseware used in the classroom is selected in one of three ways. One approach begins with the identification of weak or boring lessons and attempts to find ways in which learning can be strengthened through technology. The second starts with a search for the best available courseware at the grade/ability level of the students and proceeds with efforts of "fitting" this courseware into the existing curriculum. The last, and probably the most common, is to have the software presented to the teacher by the supervisor.

Second, the documentation accompanying educational software is of very mixed quality. Sometimes the manufacturer supplies excellent ancillary methods and materials for use. Very often, however, the suggested grade levels are too wide in range, the instructions are incomplete and the supplementary materials, if any, have little relevance to the instruction on the software. In any event, the teacher should plan not to rely on manufacturer-supplied literature.

Third, it is not the function of the teacher to do a formal evaluation of the software. It is necessary that the teacher be conversant with the software before an attempt is made to use it in instruction but the selection may be based upon cataloged or in-house evaluation forms filed with the courseware.

Fourth, the teacher must learn to focus clearly upon the measurable behavioral outcomes of the experience. Children often fail to learn because they are unable to define what is expected of them and teachers often fail to teach because they themselves are not clear exactly what they are hoping to have their students achieve.

Fifth, the appropriate placement of the software experience within the learning unit is crucial. The teacher must be very precise about where, within the greater scheme of learning, the courseware is most valuable. To do this, a list of readiness and follow-up activities should be written.

Last, brevity is a key to a successful plan. As the teacher prepares increasingly succinct statements, sharper focus is given to the objectives and sequence of the lesson.

The BARUCH Model

In response to these factors culled from the experiences of my students, and the work by Heinich, Molenda and Russel(1985), I developed the BARUCH model for creating lesson plans for teaching with electronic technology:

(B)Begin with a goal: write a general statement about the purpose of the lesson or unit.

(A)Analyze learner situation: age, grade, sex (if pertinent), number in group, learner strengths to be capitalized upon, learner weaknesses to be overcome.

(R)Redefine goal as behavioral objectives: list a series of measurable behaviors student should perform as an outcome of this lesson.

(U)Use instructional materials: list learning resources (for software, state name of program, name of package, computer, memory, cost, special needs), specify readiness or preparatory activities, delineate student participation.

(C)Complete lesson with evaluation: if you cannot measure cognitive learning, it probably has not taken place.

(H)Highlight presentation with enthusiasm! Keep all statements precise.
Avoid judgements, commentary, anecdotes, and "I"s.

Great care has been taken to assure that this procedure is neither a substitute form for evaluating software nor a guideline for evaluating the educational merit of a given piece of software. It is a tool to be used to make computing an integral part of the learning process. Figures One and Two are lesson plans created in the spring 1986 semester. While there were many excellent lesson plans prepared by the students and implemented in their classrooms with almost uniform success, I selected these two to illustrate that the model works equally well across the curriculum and the grades. The * next to an item in the figure indicates an electronic technology issue.

Figure One was developed as a cooperative venture by the teacher/students who voted situational simulations (Alessi & Trollip,1985) sometimes called simulation games, which combine the role-playing and reality modeling of a simulation with the rules and goals of a game (Heinich, et.al.,1985), the category of CAI that held the greatest promise for future educational development. The textbook used with the software is King, Rudman and Leavell, 1979.

Figure Two, written by Juan Mendez who teaches foreign languages at Stuyvesant High School in New York City, is the result of a long and tedious search for software that would truly enhance the learning experience of the gifted high school students in his classes. The lesson begins with concepts presented in class, reinforcement is provided through the software and evaluation is conducted with textbook exercises. The very first time he implemented this model, he was delighted with the flow of the new technology as opposed to the time-worn textbook teaching approach.

Each set of plans was repeatedly refined: the teachers were required to define and delineate the learning objectives and evaluation with the caveats, "If you can't measure the learning, it isn't taking place." and "Never lose sight of the size of the lesson that would be reasonable for your pupils.". They had to plan for teacher-supplied instructions where the software fell short and to assess the degree of active practice supplied by the software. In some cases, they had to replace the software when it proved to be deficient educationally or technically. Toward the end of the semester, Rosenshine (1986) published a synthesis of research on explicit teaching. The teachers were delighted to observe that their work preparing lessons with integrated software components was consonant with the findings of the research on the effectiveness of systematic, step-by-step instruction. In addition, they expressed considerable pleasure with regard to the implementation of their lessons, at the "fit" within the curriculum and positive acceptance by their pupils. Needless to say, I was the most pleased of all.

References

Alessi, S. and Trollip, S. (1985). Computer-Based Instruction. NJ: Prentice Hall.
Bonner, P. (1985). Does computer-assisted instruction work? Computers and Electronics, 23(3),43-47,92-93.
Bork, A. (1985a). Personal computers for education. NY: Harper and Row.
Heinich, R., Molenda, M. and Russel, J. (1985). Instructional media and the new technologies of instruction. Second edition. NY: John Wiley.
King, F.,Rudman, H. and Leavell, L. (1979). Understanding Our Country. IL: Laidlaw Brothers.
Lathrop, A. and Goodson, B. (1983). Couseware in the classroom: selecting, organizing and using educational software. CA: Addison Wesley.
O'Shea, T. & Self, J. (1983). Learning and teaching with computers. NJ: Prentice Hall.
Rosenshine, B. (1986). Synthesis of research on explicit teaching. Educational Leadership, 43,(7),70-77.

FIGURE 1: Fifth Grade Social Studies

BEGIN WITH A GOAL:

To allow students (Ss) to vicariously experience crossing the untamed territory from Independence, Missouri to Oregon Country by wagon train.

ANALYZE LEARNER SITUATION

Both sexes, average learners, second half of fifth grade.

STRENGTHS:

*1. Previous experience with Apple II software.
2. Completion of section 1: Expanding to the Pacific in chapter 16: Our Country Continues to Develop (King, Rudman, Leavell, 1979).

WEAKNESSES:

1. Inability to appreciate hardships of wagon-train travel.
2. Lack of experience in life/death decisions.
3. Poor map skills.

REDEFINE GOAL AS BEHAVIORAL OBJECTIVES:

*1. To build and test a mental model of crossing the trail.
*2. To make decisions of life/death consequence.
*3. To identify the states along the path of the trail.
*4. To budget money wisely.

USE INSTRUCTIONAL MATERIALS:

1. Overhead transparencies:
   a. Map: Trails to the Far West.
   b. The Oregon Trail: introduction to the program.
   c. The Oregon Trail: student worksheet.
2. Student worksheets, one per pair of Ss.
*3. Sixteen Apple IIEs in the Apple lab. (Ss work in pairs and may speak as they work.)
*4. MECC (Minnesota Educational Computing Consortium) Elementary Volume 6, program 3, Oregon, 1981.
5. Progress chart: team positions per session.

READINESS:

1. Traditional coverage of unit in social studies text.
2. Ss are encouraged to share TV or movie experiences dealing with overland travel in the mid-nineteenth century.
3. Work with transparencies should precede software use.

STUDENT PARTICIPATION:

1. Ss will name the states on the map projection.
2. Ss will complete the worksheets in pairs by agreement.
*3. Ss will load the software without teacher assistance.
*4. Ss will cross the trail keeping a written progress log.
5. Ss will report end positions to classroom teacher.
6. Ss will discuss the decisions they made and the consequences of these decisions.

COMPLETE LESSON WITH EVALUATION:

*1. Ss will travel a greater number of miles at each try.
2. Ss will write a short essay on the Oregon Trail.
3. Ss will name the states in the middle and north west.

HIGHLIGHT PRESENTATION WITH ENTHUSIASM:

1. Teacher (T) should share own experiences with program.
2. T should minimize competition and stress progress.
3. T should provide encouragement and guidance.

FIGURE TWO: Foreign Language, High School
(Written by Juan S. Mendez)

BEGIN WITH A GOAL:

To help students (Ss) master some tedious aspects of Spanish grammar.

ANALYZE LEARNER SITUATION:

Both sexes, superior learners, grades 9 and 10, 34 pupils/class.

STRENGTHS:

1. Ss have learned the names and meaning (translation) of the object pronouns.
2. Ss can identify the formal and familiar commands.

*3. Ss are comfortable with educational software.

WEAKNESSES:

Ss are unable to properly place a direct or indirect object pronoun in a sentence.

REDEFINE GOAL AS BEHAVIORAL OBJECTIVES:

1. Ss will place direct and indirect object pronouns appropriately in sentences.
2. Ss will write properly structured sentences in Spanish that include direct and indirect object pronouns.

USE INSTRUCTIONAL MATERIALS:

*1. Room with 34 IBM-PCs is available to Ss.
*2. Software: Spanish Grammar Computerized I and II.
*3. Publisher: International Software, $100.

READINESS:

Before using the software, the teacher should review formal commands, familiar commands, direct objects, indirect objects, list of object pronouns.

STUDENT PARTICIPATION:

*1. S gets software from teacher and powers-up program.
*2. S selects exercises and quizzes from menu using arrow and ENTER key.
*3. Ss read examples from screen and keys answers.
4. Ss choose to work individually or in pairs.

COMPLETE LESSON WITH EVALUATION:

* Ss will complete the worksheets taken from pages 140 and 141 in their texts with a minimum of 80% accuracy.

HIGHLIGHT PRESENTATION WITH ENTHUSIASM:

*1. S may work at his own pace.
*2. S may spend extra time with software during prep period.
*3. Exercises are lively and provide good feedback.

EDUCOM SOFTWARE INITIATIVE—UPDATE AND OPPORTUNITIES
EMPHASIS: THE EDUCOM CODE

Steven W. Gilbert
EDUCOM
Princeton, NJ

Diane Balestri
Princeton University
Princeton, NJ

ABSTRACT

The EDUCOM Software Initiative is designed to facilitate movement toward our vision of a vigorous market for software in higher education.

We have divided the Initiative into six areas:

- —Academic Software Development
- —Academic Software Awards
- —Software Information Resources
- —Institutional Contracts and the EDUCOM Code
- —Managing Software Acquisition and Support
- —Demand of Software in Higher Education

This presentation will emphasize a description of an effort undertaken jointly by EDUCOM and ADAPSO to provide guidelines for the appropriate use of software in higher education.

SPONSOR: EDUCOM

SPOTLIGHT ON BUSINESS AND INDUSTRY

Joyce Currie Little, *Chair*
Towson State University
Towson, MD

Lynne M. Davis
Ted Kahn

ABSTRACT: Seeding Expert System Technology in Industrial Settings

Lynne M. Davis, IBM, Inc., Research Triangle Park, NC

Experiments in expert systems demonstrate how to get started. Applications include financial and documentation prototype expert systems that are now used as classroom material to provide instruction in the technology. The rationale for choosing the application subject matter, the development of the knowledge bases, the test results, and the lessons learned from the projects will illustrate current experiments in expert system technology.

ABSTRACT: Computer-Based Learning in Corporations: Challenges and Opportunities for Creative Lifelong Learning

Ted Kahn, Picodyne Corporation, Portola Valley, CA

This presentation will highlight some key developments in the history of computer education over the past thirty years which have had dramatic payoffs in improving learning effectiveness, performance, and motivation in day-to-day corporate life. It will also highlight career opportunities for teachers and educators who may wish to apply their skills in this environment.

ADMINISTRATIVE USE

Lucky J. Abernathy
Gerard Kiernan, Arnold Koltun, George Psihountas, Edward Schwartz
Sandra Turner
Mark O'Shea, Howard Kimmel

ABSTRACT: Computerized Adaptive Testing

Lucky J. Abernathy, The College Board, 45 Columbus Ave., New York, New York 10023

This presentation will describe and demonstrate a new computerized adaptive test, developed by the College Board in conjunction with the Educational Testing Service. This technique combines the personal computer with newly refined concepts of item response theory (a three parameter model) to individualize the set of items presented to each student. The tests are designed to identify student entry skill level in reading comprehension, writing, arithmetic and elementary algebra.

The session will provide an opportunity to review the reading and writing components of the computerized placement tests and to experience this testing by computer process. There will also be a discussion related to the application and use of and the institutional experience with this new assessment technique.

ABSTRACT: Manhattanville College Expert Academic Advisor-Preliminary Report

Gerard Kiernan, Ph.D.; Arnold Koltun, Ph.D.; George Psihountas; Edward N. Schwartz, Ph.D., Manhattanville College, Purchase, New York 10577

We will report on an on-going project to develop an expert system that gives advice on courses to be taken to satisfy a student's major requirements and on the experiences of four novice expert-systems builders learning to become knowledge engineers.

We decided that the expert system should run on IBM PCs and that the systems developed should run as stand-alone turnkey systems. After a survey of the literature on expert-system development, we chose to use a frames-based expert-systems shell that could be enhanced with code written directly in Prolog. The system that we currently have running gives advice to mathematics and computer science majors. We will describe the knowledge acquisition and system-building interaction that is required for the development of this type of academic application. We will discuss:

- Modular development;
- Development of course recommendation trees;
- Development of taxonomy files;
- Translation of course recommendation trees into a rule base;
- Programming the expert system to work with these rules;
- General rules and mop-up rules;
- Relationship of the order of the rules in the rule base to order of the questions asked by the system;
- Testing and debugging the system; and
- Problems of integrating this approach into the traditional system of academic advising.

We will also discuss our current plans for future expansion of this system.

ABSTRACT: The School Board: An Electronic Bulletin Board for Educators

Sandra V. Turner, National College of Education, Evanston, Illinois 60201

Telecommunications is one of the newest frontiers in computer education, yet not many teachers or students have an opportunity to learn about and use telecommunications. In the Chicago area, the Computer Education Department of the National College of Education and the Education Division of Video, Etc., a local computer store, have cooperated to establish a free, electronic bulletin board system (BBS) for educators. Known as "The School Board," its purpose is: a) to provide a forum for teachers in the Chicago area to share information about using computers in their schools, and b) to give classroom teachers and their students an opportunity to become familiar with telecommunications at minimal cost.

During the first six months of operation, The School Board has received more than 1,200 calls from 200 different computing using educators. Teachers are able to get help with technical problems, share software reviews, learn about new products, find out about local users groups, and share teaching ideas. Teachers have also used the board for hands-on demonstrations of telecommunications in their classes. There have been benefits for the sponsors, too. The Computer Education Department at NCE stays in closer touch with adjuncts at extension sites and with distant student advisees; and Video, Etc. has built goodwill and raised its visibility within the local educational community.

In this session we will share how we established and maintain the BBS with minimal funding, how it is being used by classroom teachers and their students, and the factors that have been critical to its success.

ABSTRACT: The New Jersey Science Supervisors' Electronic Conference

Mark O'Shea, Fairleigh Dickinson University, Teaneck, New Jersey 07666, and Howard Kimmel, New Jersey Institute of Technology, Newark, New Jersey 07102

"Networking"--developing and maintaining communications among a group with common interests--has become a catchword in education. Conventional modes of information exchange--telephone, mail, meetings, newsletters, etc.--have failed to ameliorate teacher isolation from their peers. Electronic networks have proliferated because they provide the means for bringing people together in an unobtrusive manner that is suited to the demands of structured teaching schedules.

Computerized (or electronic) conferencing does not provide the personal touch of traditional meetings. On the other hand, the costs of removing people from the workplace in order to meet at the same time are prodigious. The Electronic Information and Exchange System (EIES), housed at the New Jersey Institute of Technology, is a sophisticated communications system that can be accessed via phone lines by anyone with a computer terminal and a modem. Its features include: electronic mail, continous conferences with a permanent transcript of the proceedings, and word processing for individual or group document composition. With this system, time, distance and cost impediments to group communications are minimized. People can communicate at the time, rate and place most convenient to them. Other significant advantages to electronic communications are search, retrieval and document organizing facilities.

The New Jersey Science Supervisors' Association has been utilizing a special conference for its members available on EIES. Several discussions important to science supervisors are conducted simultaneously in the conference. New members scroll through the conference transcript, adding their comments at any point. An analysis of the developing transcript has identified topics of greatest interest to these professionals:

- The right-to-know law and OSHA regulations
- Contact lenses in the school laboratory
- Procedures for conducting classroom observations
- Current textbooks and curriculum developments
- Laboratory safety
- NJSSA meetings and issues
- The posting of available teaching positions and candidates to fill them.

The New Jersey Science Supervisors' conference is just one example of using computerized communications for the improvement of elementary and secondary education. Additional applications are surveyed in this presentation.

TECHNOLOGY IN THE FOREIGN LANGUAGE CURRICULUM:
A RESOURCE FOR TEACHERS

John A. Vaille
Technology in the Foreign Language Curriculum Project
Modesto, CA

ABSTRACT

Expanding the Communicative Experience through Technology.

Foreign language teachers are using technology to bring foreign cultures into the classroom, making it possible to extend students' learning experiences beyond those confines as never before. Technology is providing many students with a new set of tools that enables them to explore foreign languages and cultures by placing large bodies of current information at their fingertips. To an extent never before possible, today's technology facilitates this exploration by providing many means for the student and teacher to gain access to, experience, and use a language.

Foreign Language Instruction and the Roles It Can Play in the Language Learning Experience

Communicative technology-assisted language learning is the means by which educational technology can be applied effectively to the foreign language learning process. A communicative technology-assisted language-learning-system can assist the foreign language teacher in confronting some of the challenges of providing effective instruction. In today's communication-oriented classroom, communicative technology-assisted language-learning systems can provide the following:

- Authentic cultural and language experiences as a shared communication referent for the learners;
- Direct stimulus for communicative interaction;
- A means of receiving and transmitting written and oral messages, especially among people separated by time and/or distance; and
- Instructional assistance during periods of multitask activity.

In delivering these means to the foreign language program, a communicative technology-assisted language-learning system can assume a number of roles in today's teaching/learning process. An examination of these roles illustrates the ways in which this system might interact with a learner:

- As Instructor - Communicative technology-assisted language learning can be used to help students develop proficiency in communication and master necessary grammatical structures,vocabulary items, and other skill components when necessary.
- As Learning Partner - Communicative technology-assisted language learning can provide cues and other stimuli to interaction. It becomes communicative partner for the student.
- As Facilitator - In this capacity, these systems serve mainly as a means by which students can communicate. For example, the student may use a word processor to create a story, essay or speech in the foreign language.
- As Data Source - Technology can provide students and teachers access to information: written, pictorial or auditory; which can serve as raw material for communicative exchanges or cultural experiences.
- As Assessor - Technology can serve to evaluate foreign language performance.

The Technology in the Foreign Language Curriculum project was created to develop the link between the California foreign language curriculum and the technology that may serve it. The Resource Guide it has produced brings the issues in communication-based foreign language instruction and technology's role in that process into sharp focus; it provides models of the use of technology that enhances foreign language learning, and recommends microcomputer software and instructional video programming that can assist foreign language teachers in accomplishing their goals. It is the project's aim that this Resource Guide should provide a point of departure for the individual foreign language teacher's own implementation of technology in the classroom.

This presentation will focus on the issues raised above, demonstrate technology discovered in our investigations, and describe the criteria applied in selecting technology for inclusion in the Resource Guide. Information regarding the availability of the Technology in the Foreign Language Curriculum Resource Guide will be supplied to all participants.

ICCE CERTIFICATION RECOMMENDATIONS FOR COMPUTER-USING EDUCATORS

Tony Jongejan, Chair
Western Washington University
Bellingham, WA

ABSTRACT

The proliferation of computers in schools has resulted in a need for knowledgeable computer-using educators. All educators need to know how the computer can be used as an instructional tool for themselves and their students to help teach/learn class material. In addition, they need to be aware that the computer can be a personal tool for classroom preparation and management.

There is strong evidence that curriculum content must change to reflect the changing nature of the marketplace and take into consideration computer based intellectual and physical aids to productivity in our society. Thus, all educators must also be ready to adapt to these changes the computer will bring to their classes. Also, knowledgeable educators are needed who are ready to teach courses/units where the computer is the focus of study.

All these factors have resulted in a need to establish certification requirements for educators currently entering the teaching profession. This means that competencies and course requirements need to be established for the following groups:

1. All teachers;
2. Teachers by grade level or subject;
3. Computer science teachers; and
4. Computer specialists (coordinators, etc.)

In the category of "all teachers," there is a core of knowledge and skills independent of grade/subject area with which all teachers should be competent. These are based on a three-part model. That is, all future teachers should be provided with a learning experience that deals with:

1. The computer as a system;
2. Computer applications in education; and
3. The computer as a tool for personal use.

After the introductory experience for all education students, additional content should be integrated into all required and optional courses in the teacher preparation program. These experiences should match the specific program needs of the individual students followed by knowledge and skills specific to individual grade level/content area.

Educators who are going to teach courses/units where the computer is the focus of study need additional formal training. Computer science teachers need formal training in computer science. Recommendations for this training have been specified by the Association for Computing Machinery (ACM) and endorsed by the International Council for Computers in Education (ICCE) Teacher Education committee.

Finally, computer specialists (coordinators, etc.) need different skills, knowledge, and perspective. They need supervision, curriculum development, and administrative skills in addition to a broad forward-looking perspective on the role of computers in their school and/or district.

ICCE believes that recommendations for the curriculum of teacher education programs and for certification requirements should be a grass-roots development with practitioners formulating these recommendations.

Individual states are beginning to address the issue of teacher certification, and some states have legislation in progress. The ICCE committee has been studying the issues involved and developing recommendations for all groups of computer-using educators. The committee's goal is to formulate recommendations and guidelines for state committees to use in developing their own teacher certification requirements and for Schools of Education to use in developing their computer education curriculum.

This panel, which includes members of the ICCE committee, will discuss the process of establishing certification requirements, report the committee's progress, and look at the progress of certification nationwide.

PANELISTS

Dave Moursund
International Council for Computers in Education
Eugene, OR

Bobby Goodson
International Council for Computers in Education
Sunnyvale, CA

Bonnie Mathies
Wright State University
Dayton, OH

SPONSOR: ICCE

APPLICATIONS

Robert Aiken, Douglas Peterson, Ann Commito
David Arney, Anthony Manganiello, William Rue
Ernest Randolfi
Ann Hill Duin

ABSTRACT: A Computing Survival Skills Course for Liberal Arts Colleges

Robert J. Aiken, Assistant Professor; Dr. Douglas L. Peterson, Associate Professor; Ann E. Commito, Assistant Professor; Mathematics and Computer Science, Hood College, Frederick, Maryland 21701

Hood College has realized the need to develop a "Computing Survival Skills" course that will provide the undergraduate liberal arts student with a combined mathematics/ computer science requirement in basic computer literacy and the handling of numeric data. This course (core) is intended to provide students with computational skills from two perspectives; mathematical analysis and interpretation of numeric data, and basic techniques of using computer hardware and software.

The two approaches will be integrated into a final project in which the student will use a personal computer (IBM PC) and software packages (a word processor, a spreadsheet, a spelling checker and a graphics generator) to produce information and materials for a written report and a verbal presentation at the end of the semester. The report topic will be one that is suitable for data gathering and/or for analyzing preexisting data so that the student will learn to argue a point using basic statistical techniques while learning the use of computer tools to help him/her analyze the data and prepare the results for presentation. The subjects chosen can be from any discipline as long as it meets the above criterion and is not so complex as to inhibit completion of the project.

This is a course, unlike other introductory courses that stress the use of procedural languages, that integrates basic knowledge of computer software packages, verbal, written, and statistical skills on a group basis to accomplish a goal.

ABSTRACT: Implementation of Micro computers in Mathematics Courses at The United States Military Academy

David C. Arney, Anthony J. Manganiello, William P. Rue, Department of Mathematics, United States Military Academy, West Point, New York 10996

With the recent growth of microcomputer technology, revolutionary changes have been introduced in many facets of military operations. To ensure that the Army will be receiving officers fully prepared to be leaders in this era of emerging technology, the United States Military Academy had dedicated itself to a curriculum that emphasizes computer proficiency. This year, in a major step toward fulfillment of this goal, each member of the entering freshman class received an IBM-AT-compatible Zenith Data Systems Z-248 personal computer. Within four years, under this plan, every cadet in the 4,400 student body will have similar computers at their desks and in their classrooms, thereby ensuring that instruction and classroom support are compatible with the cadet hardware.

This proliferation of computer hardware provides an opportunity for the Academy to effectively incorporate computers into the entire curriculum. During this academic year, the Academy began curriculum adjustments to achieve its goal. The Department of Mathematics is leading the way in the use of microcomputers in courses during the current academic year (1986-1987) and is making plans for more curriculum adjustments in the future.

This presentation will report on the integration of the microcomputer in the mathematics courses at the Academy. Much of the work in the Mathematics Department has been centered on development of computer based applications for the standard calculus courses. Various off-the-shelf products and in-house developments are being studied and tested in the classroom. The Mathematics Department is attempting to develop a complete program that makes effective use of the microcomputers thereby enabling graduating cadets to use their knowledge of computers throughout their military careers.

ABSTRACT: A Survey of Microcomputing Courses for Health Educators

Ernest A. Randolfi, Ph.D., Department of Health and Physical Education, Syracuse University, Syracuse, New York 13244

Questionnaires were mailed to 220 colleges and universities in North America that offer graduate degrees in health education to determine the extent of educational computer instruction offered within these programs. Of the administrators who responded (n = 120 usable surveys), 89.6 percent indicated that all health education students should receive instruction in the use of microcomputers in health education.

Telephone interviews were conducted with individuals at institutions that offered or planned to offer a microcomputing course within the next year (n = 31). Six programs required that all health education students complete the microcomputing course. A few programs offered more advanced courses for students that had completed an introductory course. The majority of courses covered a range of topics including hardware operation, programming, word processing, spreadsheet operation, health education tutorials, simulations, and assessments.

A trend away from teaching programming was identified by survey participants. They also indicated an increasing emphasis on the appropriate use of health promotion applications. A few individuals reported campus restrictions regarding the type of computer course that could be offered. Frequently the focus of instruction was found to coincide with the type of health educator being prepared, i.e., school, community, public, or allied health educator. Recommendations concerning microcomputing courses for health educators include offering a course at the undergraduate level focusing on use of technological applications, and a course at the graduate level emphasizing the diffusion of technological innovations in health education.

ABSTRACT: Transforming On-line Computer Assisted Instructional Displays: An Investigation of Two Audience Analysis Programs

Ann Hill Duin, Technical Communication Program,
University of Minnesota, St. Paul, Minnesota 55108

On-line information is often merely printed text and instructions transferred verbatim to the on-line display. Because on-line information is so new, we instructors have not immediately known where to get help to create aesthetically pleasing and reader-useful displays. Research on document design can offer guidelines in this area.

After researching document design, I developed a list of guidelines for instructors to follow when developing on-line instructional displays. In order to investigate the effectiveness of these guidelines, I revised a computer assisted instructional program on subject and audience analysis according to the guidelines. I then tested the two versions (original vs. revised) with junior and senior college-level students enrolled in a preprofessional course entitled Writing in Your Profession. Three forms of data were kept: observational logs, objective measures (time to complete the exercise, amount of help needed and types of questions asked, quality of resulting audience analysis work), and subjective measures (questionnaires, interviews). A repeated measures analysis of variance was used for the objective measures, and Chi-Squares were performed on the responses to the questionnaire items.

When discussing this study, I will concentrate on those visual, conceptual, and linguistic revisions that proved most effective in terms of increasing the program's readability and usability.

MOVING TOWARD CURRICULAR INTEGRATION: MERGING SOUND INSTRUCTIONAL PRACTICES WITH APPROPRIATE AND EFFECTIVE COMPUTER SOFTWARE

Lynn M. Lieberman
Computing and Technology
Oakland Schools
Pontiac, MI

ABSTRACT

As school districts move toward using the computer as part of the instructional process in existing curricular areas, computing coordinators and curriculum directors have expressed a need for direction and suggestions of how to integrate "tool" and instructional software packages into classroom lessons. As a result, the State of Michigan has undertaken the following activities to address computer integration:

1. Development of a monograph that includes strategies used by classroom teachers in elementary language arts and secondary mathematics, social studies, science, and music, as well roadblocks that must be faced and resources that are necessary. This product resulted from a forum for computing coordinators at which five teachers shared their "success stories" of how they are integrating computing into their respective subjects;
2. Identification by survey of how districts are planning for and approaching integration;
3. Creation of a model for the development of instructional lessons that incorporates what we know about the instructional process; and
4. Development of instructional lesson plans that are based on the model.

Initiated at Oakland Schools, the purpose of the lesson development project was to develop immediately usable instructional lesson plans that provide ideas and activities for using appropriate software packages in instructional units. The project objectives were to: (1) design a model that incorporated effective instructional practices, (2) examine curriculum and exemplary software and determine a fit, (3) use the model to create lesson plans that focused on curricular topics and objectives, demonstrated how quality software could be used within the lesson, and required little planning or modification, and (4) implement the lessons.

The model is adopted from the effective instruction lesson design [known by various names, one being Instructional Theory Into Practice (ITIP)]. The lessons and handouts that have been developed from this model provide the necessary information that teachers need to plan instructional activities. Software used in the "model lessons" approached strategies for problem solving, cooperative learning, information gathering, and the higher-order thinking skills involved in applying gathered facts.

During the lesson development, the writers examined the curriculum that is currently being taught in the schools and looked at available software to determine where there was a "fit." After choosing a topic and appropriate, positively evaluated software, the authors then narrowed the topic to the specific objective(s) that would be covered in the lesson and determined where the software might be appropriately used. The writers then focused on the instructional process, recognizing how topics are taught and applying effective instructional strategies and classroom management considerations.

The state Technology in Michigan Education (TIME) project has adopted this model for the development of curricular materials at five regional centers. The South-Eastern Michigan Technology Education Consortium (SEMTEC), which includes Oakland County and seven others, is one of these regions that is currently developing instructional materials that will be field tested in classrooms in the area. At the training session held in December, SEMTEC project participants were introduced to the model, reviewed the elements of lesson design, and worked in small groups to design a lesson based on the model. Although participants commented that the process of creating a lesson for others to use required effort, a great deal of time and careful thought, the reaction to the planning format was quite favorable.

As district curriculum planning committees review existing curriculum and make recommendations for revisions and additions to the content, the use of all forms of instructional technologies should be considered for their appropriateness as a means of addressing curricular goals and objectives. It is hoped that these instructional lessons which incorporate effective teaching practices with the use of quality software, will demonstrate how this technology can be effectively integrated into classroom experiences.

COMPUTER IMPLICATIONS AND ETHICS: A FIRST BUT LAST PRIORITY

George G. Bear, Ph.D.
Department of Educational Studies
University of Delaware
Newark, DE

Abstract

In this paper the relationship between definitions of computer literacy espoused by authorities in the field of computer education and what typically is taught in the computer literacy program is examined. Several research studies are reviewed that support the argument that while the understanding of computer implications and ethics is viewed as a first priority in most definitions of computer literacy, it becomes the last priority when computer literacy is actually taught. Reasons why the teaching of computer implications and ethics is often the last priority in the classroom are explored. These reasons include: (1) restricted perceptions of computer literacy and computer ethics, (2) problems measuring objectives related to computer implications and ethics, (3) lack of trained staff, and (4) lack of teaching materials.

Computers are now firmly entrenched in the schools. Consequently, most educators no longer debate the need for computer education, but instead are asking the more important question, "What should schools teach children about computers?" The most common response, which unfortunately falls far short of adquately answering the question, is that computer education should prepare students to be "computer literate." Here is where debate now prevails.

As noted by Deringer and Molnar (1982) "The term 'computer literate' has many different definitions. Many or most of these differences are a matter of perspective. Some define computer literacy as it is taught; some see computer literacy as it should be taught; and some visionaries see computer literacy as it could be taught given the sophisticated technology predicted to be soon available . . . to some a general awareness of computers is sufficient; to others, a technical skill that can only be acquired by hands-on experience is mandatory; to still others, students must learn to write programs that do things -- solve differential equations or create poetry." (p. 5)

The term "computer literacy" first appeared in the late 1960s and received national attention in 1972 when the Conference Board of the Mathematical Sciences Committee on Computer Education suggested that all children should complete a junior high school computer literacy course to learn about computer appplications and capabilities. Although various definitions of computer literacy were offered in the 1970s, it was not until the microcomputer was first introduced, in the early 1980s, that educators across the nation began to closely examine the meaning of computer literacy. Differences of opinion quickly emerged, largely over the level of programming skills that students should be taught. Some computer educators perceived computer literacy skills as being communication skills, analogous to reading, writing, and speaking. A person would need to know programming in order to communicate via computer. Others viewed computer literacy as being a matter of knowing about computer applications and implications, but not necessarily programming (Anderson, 1982).

The computer literacy debate continues. Advocates of the "computer literacy equals computer programming" view believe that students should be taught programming skills because (a) students are likely to use them throughout their lives, (b) programming teaches good logical thinking skills, and (c) many careers require programming skills. Opponents of this view argue that although some students should learn programming, such as students planning to enter careers in science or math, the majority of students do not need to learn programming, especially BASIC. Foremost among the critics of the programming equals computer literacy position is Alfred Bork. Bork (1985) gives five excellent reasons why the majority of students should not be taught programming: (1) they will never use programming skills after finishing high school since inexpensive written programs will be readily available for almost all areas of application, (2) the skills that they do learn are likely to be obsolete by the time they graduate, (3) the wrong computer programming language is likely to be taught (i.e., BASIC, a language rarely used outside of the home and precollege setting), (4) what they are ususally taught in programming courses is likely to result in the practice of bad programming habits, and (5) there is no clear evidence indicating that programming fosters problem solving, logical thinking, or any other ability other than programming per se.

Largely in response to these and other criticisms of teaching programming in the computer literacy curriculum, it is not uncommon to find programming skills missing in popular definitions of computer literacy. If one were to review the multiple definitions of computer literacy, one would discover that the majority of the definitions emphasize computer applications and implications rather than programming skills. In fact, as indicated in Table 1, a concern about the impact of computers, and a related concern about the student's attitude toward computers, appear more frequently in computer literacy definitions than the need to learn programming skills per se.

Despite the priority that an understanding of computer implications receives in most definitions of computer literacy, computer literacy is too often mistakenly equated with BASIC computer programming when it comes to the actual teaching of computer literacy (Kelman, 1984). Forunately, this appears to be less true today than a few years ago. There is evidence that computer applications now rivals programming in terms of the amount of emphasis placed on these two major components of computer literacy (Becker, 1986). However, there is no evidence that an understanding of computer implications and computer ethics -- an equal or higher priority than programming in most computer literacy definitions -- receives sufficient attention, much less priority status, in the classroom.

The remainder of this paper examines the relationship between definitions of computer literacy espoused by authorities in the field of computer education and what typically is taught in the computer literacy program. Several research studies are reviewed that support the argument that while the understanding of computer implications and ethics is viewed as a first priority in most definitions of computer literacy, it becomes the last priority when computer literacy is actually taught. Reasons why the teaching of computer implications and ethics is often the last priority in the classroom are explored.

What defines computer literacy in the schools?

A school's definition of computer literacy should translate into educational goals, which should directly influence what is actually taught to students. In most schools a definition of computer literacy is linked to a mandate from the local school board, state department of education, or both, stating that beginning with a certain date, students are to be computer literate. A goal statement, based on a definition of computer literacy adopted by the school (sometimes without choice), is generally developed by a committee of educators appointed by the school administration. From this goal statement more specific curriculum objectives typically emerge. These objectives are used by teachers to guide them in their teaching of computer literacy. Thus, what is taught in the classroom is often dictated by how one defines computer literacy. For example, a school that accepts Luehrmann's definition of computer literacy (i.e., computer literacy is the ability to do computing), is likely to spend very little time teaching anything other than programming. On the other hand, those schools that adopt Bork's definition should cover a much broader range of important topics.

The state of Virginia is one of the many states that now require that all students be computer literate upon graduation. Computer literacy is defined by Virginia as being "the ability to understand the capabilities, applications, and implications of computer technology and to use this knowledge to function in society." Like most other popular definitions of computer literacy, the definition emphasizes applications and implications and not programming.

In Virginia, computer literate students are those who achieve 12 objectives, or outcomes, that reflect an adequate level of ability and knowledge in computer literacy. These objectives, which were developed by a state-appointed committee of computers educators and derived from Virginia's definition of computer literacy follow:

1. The student will define selected computer terminology.
2. The student will identify the basic components of a computer and describe the function of each.
3. The student will load and run a prepared program.
4. The student will use the keyboard to interact with the computer.
5. The student will select and use an appropriate program to solve a given problem.
6. The student will describe a computer program.
7. The student will analyze a simple problem and describe procedures to solve that problem.
8. The student will describe the recent development of computer technology and its applications in daily living.
9. The student will describe the potential and limitations of the computer.
10. The student will identify and evaluate major ethical, social and economic issues that relate to the use of computers.
11. The student will explore computer-related careers and recognize the value of computer skills for future education and employment.
12. The student will identify electronic data bases as information sources.

Note that these objectives are similar to those of many computer literacy programs, particularly those that emphasize applications and implications but not the teaching of programming skills.

Virginia's definition of computer literacy and its 12 teaching objectives have been used by many Virginia schools in the development of comprehensive computer literacy curriculum guides consisting of specific teaching objectives and scope and sequence charts. These guides, often used in conjunction with computer literacy textbooks and other commercial materials, typically dictate what students are actually taught about computers in the classroom. Unfortunately, after translations, refinements, and additions, the original definition and objectives of computer literacy may or may not be consistent with what is taught to students. The "ability to understand the capabilities, applications, and implications of computer technology and to use this knowledge to function in society" may be replaced by "knowledge of BASIC programming" and very little knowledge of computer implications and ethics. Good intentions do not always translate into good behavior.

Although research has not specifically examined the consistency, or inconsistency, between what is taught and what should be taught in computer literacy programs, the results of several studies indicate that a serious discrepancy does exist between the two. Kessler (1986) recently surveyed 135 directors of computer education in Virginia (of which 85% responded). The survey asked whether programming skills, applications skills, or social implications were most emphasized in their school district's computer literacy program. Kessler found that 63% of the districts emphasized applications, 23% programming (twice as many taught BASIC than Logo), and 14% emphasized social implications. While the finding of a greater emphasis on applications than programming is consistent with curriculum-teaching expectations reflected in Virginia's definition and objectives, the finding that social implications was the least emphasized component reflects inconsistency.

It should be noted that directors of computer education, and not teachers, responded to the survey. Thus, we do not know if teachers place greater importance than administrators on implications and ethical issues. Likewise, we also do not know if the amount of time that is directed to teaching implications and ethics is sufficient for teaching students to be responsible citizens. The survey did not answer the question, "Is enough emphasis being placed on social implications?" nor "Are state objectives related to social implications being achieved?" Unfortunately, as suggested in the studies to be reviewed below, negative responses would likely have resulted if these two questions had been addressed.

In a recently conducted pilot study, I asked 96 students in grades 5-7 to "write a short story in which a person or persons face some kind of moral problem involving a computer or computers that is not easy to solve." The students attended a small rural school district, with a ratio of 1 computer per 17 students and a computer located in each classroom, that had adopted Virginia's computer literacy definition and objectives. The school district had also developed an exemplary computer literacy curriculum that emphasized applications and implications. The short story assignment given to the students provided a means of examining if moral problems spontaneously produced by students reflect the major ethical issues that are taught, or should be taught, in the computer literacy program. Based on their stories, 14 themes were identified. These themes were grouped into four major categories: (A) Intentional Illegal Access (cases of accessing school, government, bank, and other computers), (B) Accidental Illegal Access (cases of accessing school, government, bank, and other computers), (C) Problems in Relation to the Operations of Computers (cases of accidentally causing a computer error, or computer malfunctioning), and (D) Other (including 3 cases about how to acquire a computer, 1 case each involving cheating, computer errors, equity, health concerns, etc.). A fifth category consisted of undefined themes or themes unrelated to computers. The following percentages of students wrote about the respective themes: 26%, category A; 24%, category B; 17.7%, category C; and 20.8%, category D. The undefined category included 11.5% of the stories. Although not predicted, sex differences were found, Chi Square (4) = 11.93; $p < .05$. While more boys than girls wrote about Intentional Illegal Access (35% vs. 17%) and Accidental Illegal Access (31% vs. 17%), more girls (23%) than boys (13%) wrote about Computer Operations and Other themes (31% vs. 10%).

For the most part, the problems identified by the students were those that are addressed in computer literacy curricula that include the teaching of computer implictions and ethics. However, there is a striking absence of several important topics, including copyright issues, job displacement, computer equity, health and social issues. We do not know if such an absence resulted from these topics not being taught, being taught but not learned, or being taught and learned but being of little concern or interest to the students.

Results of a larger study by Gabriel (1985) indicate that students know little about computer ethics and a major reason is that computer ethics is seldom taught. In a validation study of a computer literacy test, Gabriel found that students enrolled in computer literacy and computer science programs answered approximately 50% of the test items correctly. Gabriel concluded that one likely reason for their low performance was that the students were taught little more than programming skills. In support of this conclusion, Gabriel examined the computer curricula of 115 schools and discovered that the average fourth grader in the study was exposed to only 2 of 13 computer literacy objectives, while the average high school student was exposed to only three computer science objectives. The use of computers, BASIC programming, and hardware operations were the objectives that were typically taught. Although seventh graders and high school students enrolled in computer literacy courses covered slightly more objectives, four and eight objectives respectively, the emphasis was still computer use and programming.

We are confronted with an enigma in computer education. Although most definitions of computer literacy indicate that computer implications and computer ethics should be a higher priority than programming, evidence suggests that in many classrooms the teaching of programming is the first priority, while the teaching of computer implications and ethics is the last priority. Several possible reasons, or barriers, can explain this discrepancy as well as why the teaching of computer implications and computer ethics is delegated to last priority.

1. Restricted perceptions of computer literacy and computer ethics. As previously noted, many schools equate computer literacy with computer programming. Most often this is because "everyone else is doing it." In these schools the single goal of the curriculum is to produce students who can program computers. When students are not programming on the computer, they are developing flowcharts or working on programs at their desks while waiting for access to a computer. Typically, students are assigned to read a few sections in the text that address confidentiality of information, computer applications, and computer careers. However, little, if any, time is spend dealing with social and ethical issues. This is largely due to either the school's restricted perception of computer literacy, or the belief of the teacher, school administration, or community that ethics, values, and controversial issues should not be addressed at school, but only at home or church.

2. Problems measuring objectives related to computer implications and ethics. It is much easier for a teacher to have students demonstrate their knowledge of specific programming skills than to demonstrate their understanding of social implications and ethical issues. It is even more difficult to measure whether a computer literacy curriculum has any affect on the student's attitudes, values, or moral reasoning. Moreover, when changes in the affective domain are evidenced, we do not know if such changes will have any effect on the actual behavior of the students when they are confronted in real life with computer-related ethical issues and problems. Consequently, schools that stress measurable student progress

spend little time in the classroom on topics related to attitudes, values, morals, or controversial issues. When such areas are covered in the computer literacy curriculum, course objectives are frequently limited to specific computer literacy laws that students are asked to recognize or recall, but not necessarily understand. Educators should not expect to have any lasting influence on the sociomoral development of students if they do not teach them to comprehend the meaning and intent of laws, as well as the various perspectives that relate to the issues involved.

3. Lack of trained staff. Numerous national and state committees have been assigned the task of studying the educational needs pertaining to computer literacy education and nearly all have identified the lack of trained staff to be the foremost barrier to the successful implementation of computer education programs. The same concerns that are involved in the debate over the concept of computer literacy for students also apply to the issue of teacher training. What skills must a teacher possess in order to be computer literate? To teach computer literacy? Are only programming skills necessary or should a school district spend valuable inservice training time to teach about the "nonprogramming" aspects of computer literacy? Because many schools have a narrow conceptualization of computer literacy and/or prioritize the need for teaching programming, education is currently faced with a shortage of teachers who are adequately trained to teach about the social impact of computers. Teachers without adequate training in computer programming, even when asked only to introduce students to the concept. Consequently, they are less effective than teachers with a firm background in computer programming. The same holds true with the teaching of ethics and the social impact of computers -- those teachers without adequate training in the teaching of computer ethics and social impact are likely to feel uncomfortable and to be ineffective in their teaching of these important components of computer literacy.

The lack of teachers with sufficient training is the most critical barrier to the successful teaching of computer literacy. This is particularly true in the teaching of computer ethics because in all likelihood, the other three barriers would not exist if educators understood the importance of these dimensions and possessed the skills to teach them.

4. Lack of teaching materials. There seems to be an endless number of textbooks, workbooks, and related materials on programming, and the teaching thereof. Many popular textbooks that claim to offer a comprehensive approach to computer literacy provide the learner with sufficient skills in programming and computer applications but fall short of presenting an adequate treatment of ethical issues and the impact of computers on society. Consequently, when teachers base their instructions on available texts and materials, implications and ethics often go untaught. With few exceptions (Bear, 1986a, 1986b; ICCE, 1985), curriculum guides, teaching strategies, and related materials dealing with the teaching of the social impact of computers are clearly lacking.

Conclusion

Research of the teaching of computer implications and computer ethics is clearly needed. There is general agreement in the field of computer education that to be computer literate students should have an understanding of ethical issues related to computers. However, we know very little about the extent to which this critical component of computer literacy is actually taught and when taught, whether it is done effectively. It is also clear that barriers to the teaching of computer implications and ethics need to be addressed before educators will begin to assign the same priority to teaching students to be responsible citizens in a computer-oriented society as they now assign to the teaching of BASIC programming.

TABLE 1: WHAT IS COMPUTER LITERACY?

"Computer literacy is the ability to do computing." (Luehrmann, 1981)

Computer literacy comprises: (1) the ability to control and program a computer to achieve a variety of personal, academic, and professional goals; (2) the ability to use a variety of preprogrammed computer applications (software) for different purposes; (3) the ability to understand the growing impact of computers on our society; (4) the ability to make use of ideas from the world of computer programming as part of an individual's collection of strategies for information retrieval, communication, and problem solving. (Watt, 1981)

"Whatever understanding, skills, and attitudes one needs to function effectively within a given social role that directly or indirectly involves computers." (Anderson & Klassen, 1982)

Computer literacy consists of five primary components: (1) how computers are used in society, (2) how computers are operated, (3) how computers are instructed, (4) how computers work, and (5) how computers affect society. (Brumbaugh & Rawitsch, 1982)

"A general understanding of electronic computing: an area of knowledge that includes (a) an understanding of the technology used when processing information, (b) an understanding of the effects computers have had and will have on society; and (c) an understanding of how computers are problem-solving tools." (Hofmeister, 1983)

"Whatever a person needs to be able to do with computers and know about computers in order to function in an information-based society. . . . . For students in grades K through 8, 'computer literacy' means ability to use suitable programmed computers in appropriate ways to assist in learning, handling information, and solving problems; and the ability to make informed judgments about social and ethical issues involving computer and and communication systems." (Hunter, 1984)

"Computer literacy may be defined as whatever a person needs to know and do with computers in order to function competently in our information-based society. Computer literacy includes three kinds of competence: skills, knowledge, and understanding. It includes: (1) the ability to use and instruct computers to aid in learning to solve problems and in managing information; (2) knowledge of function, applications, capabilities, limitations, and social implications of computers and related technology; and (3) understanding needed to learn and evaluate new applications and social issues as they arise." (Advisory Panel for the National Center for Educational Statistics, 1984)

"The term computer literacy can be considered to mean the minimum knowledge, know-how, familiarity, capabilities, abilities, and so forth, about computers essential for a person to function well in the contemporary world." (Bork, 1985, p. 33)

ENHANCING MINORITY STUDENT EDUCATIONAL OPPORTUNITIES THROUGH INSTRUCTIONAL COMPUTING: A REPORT ON PROJECT MICRO

Carol Edwards
Jacqueline Jordan Irvine, Ph.d

Southern Coalition for Educational Equity, Inc.
75 Marietta Street, NW
Suite 308
Atlanta, Georgia 30303

ABSTRACT

Project MiCRO, a four year model project of the Southern Coalition For Educational Equity, is demonstrating ways of effectively using microcomputers to enhance the educational computing and critical thinking skills of minority and economically disadvantaged, seventh and eighth grade, urban and rural students. Research shows that minority students who are economically and/or educationally disadvantaged do not use microcomputers as frequently as white, affluent and/or educational advantaged students. There are also serious qualitative differences in the ways these populations use microcomputers. Project MiCRO is demonstrating that these inequities can be corrected through its work with schools in three states, Alabama, Georgia, and North Carolina. In project schools all teachers are trained to teach students to use microcomputer applications as tools to extend their information base to solve problems and make decisions in their academic subjects. In addition some students are randomly selected to take a microcomputer based critical thinking skills course. Parents are involved in Project MiCRO through outreach and parent-directed, in school microcomputer based activities. Results show that Project MiCRO students have made greater gains in educational computing than control school students. Pretest-post test results also show gains. Interview and survey data show that students, parents, teachers, and principals believe that Project MiCRO has enhanced instruction.

BACKGROUND

Project MiCRO, Minority Computer Resource Opportunity is a model development program which addresses the growing disparity between economically disadvantaged students, minority students and affluent, educationally advantaged students. The project attends to this problem in both quantitative and qualitative ways.

Project MiCRO is a program of the Southern Coalition For Educational Equity, Inc. Founded in 1978, SCEE is an organization which works to make public schools in the south effective, humane, safe learning institutions for all children. In addition to Project MiCRO SCEE has developed a school effectiveness model for middle schools which is currently being expanded into the elementary grades.

PROGRAM DESCRIPTION

Goals and Objectives

Effective education now, for tomorrow's world, can prevent the crises which will inevitably accompany a large population of unemployed and under employed workers. However without immediate intervention existing educational agencies are unlikely to successfully meet this challenge with regard to computer technology.

Project MiCRO has the following goals:

1. To develop a model program, using instructional computing, to increase the critical thinking skills of minority and economically disadvantaged middle/junior high school students.

2. To develop a model using instructional computing as a tool through which students extend their information base to help them make decisions and solve problems in all academic subjects.

The goals of Project MiCRO are being accomplished by implementing the following program objectives which, taken as a whole, form a comprehensive approach to achieving the program goals.

The strategies are:

A. To develop instructional components relevant to urban and rural minority and economically disadvantaged students in seventh and eighth grades which enable them to practice critical thinking skills through the use of instructional computing.

B. To familiarize parents with instructional technology to assist them in becoming active partners in their children's microcomputer based learning.

C. To train teachers to integrate instructional computing effectively into the curriculum.

D. To define an effective delivery system for the use of microcomputers as instructional tools, document, and disseminate that system.

Significance

By the time today's twelve-year olds enter the workplace, futurists predict that information age jobs will be of two types. For those who have good critical thinking skills, can apply technology as a problem solving tool and can adapt quickly to innovation, there will be lucrative, creative jobs. For those who use technology passively and possess only the traditional basic skills, there will be dead-end, low paying, "technology-directed" jobs. As technology becomes more sophisticated such jobs will be eliminated and those workers unemployed.

A gap between the haves and the have nots in computer technology is forming. Those who have limited access to computers are the traditional have nots in American society -- minorities, the poor, and women.

As microcomputer growth in education occurs, the access gap continues to get worse. According to Market Data Retrieval in 1984, schools with 5% or less students below poverty level on average had one instructional microcomputer for every 54 students; schools with at least 25% of their students below poverty level had one for every 73. The acquisition of computers is occurring at a faster rate in affluent schools. For example, in 1983 there was a 2.2 gap between the number of computers in affluent and poor schools. In 1984 that gap had grown to 3.8. Without immediate intervention this unhealthy access gap will continue to grow.

Minority and economically disadvantaged students have less access to computers in nonformal settings. Average income among purchasers of home computers is $35,000. The average income of African-American families is $16,786, Hispanic, $19,027, well below that of home computer owners. Computer camps are dominated by white, affluent boys.

Inequity in educational computing is a qualitative as well as a quantitative one. According to a Johns Hopkins University study minority high school students spend more time on drill and practice and less time on programming than white high school students do. Educationally advantaged students at all grade levels receive more instructional time on microcomputers than other students. While using microcomputers educationally advantaged students engage in more problem solving tasks and less drill and practice at both elementary and secondary levels. Since minority students are underrepresented in educationally advantaged groupings, such as advanced placement and gifted classes, they are adversely effected by this practice. Schools with large minority populations have fewer microcomputer using teachers than do predominately white schools. Lack of emphasis on "higher order" uses of microcomputers compounded by less overall computing time both in and out of school and less access to microcomputer using teachers create a devastating disadvantage for minority and economically disadvantaged students which is likely to have troubling consequences for their future options.

Qualitative inequities in educational computing are particularly dangerous because they are subtle. They are reinforced by institutionalized racism and sexism which are manifested in low expectations for minorities and girls. Project MiCRO is designed to address these issues.

Site Selection

Project MiCRO operates in three southern school sites, Pantego Junior High School in rural Beaufort County, North Carolina, Austin T. Walden Middle School in Atlanta, Georgia, and Paul Lawrence Dunbar Junior High School in Mobile, Alabama. These sites, two urban and one rural have large minority student populations with little or no access to microcomputers. Microcomputer usage prior to Project MiCRO was limited to remedial drill and practice.

Sites were selected which met the following criteria: 1) large minority population; 2) declining income base; 3) current and projected stability; 4) support of school board and superintendent; 5) "Effective Schools" characteristics; 6) positive attitude toward innovation; 7) history of parental involvement; 8) teacher with knowledge of a programming language; and 9) willingness to provide resources. Meetings were held with the faculties of all potential sites as part of the selection process. The faculties of the chosen schools expressed a strong desire to participate in the project.

Using the Microcomputer As An Information Extending Tool

Computer literacy means different things to different people. This project has developed its own working definition of computer literacy which is reflected in all program activities. In Project MiCRO effective instructional use of microcomputers results when the students use them to extend their information base to solve problems and make decisions. Computers are used much as they are by adult managers in business and science. The major difference is that the problems solved and decisions made relate to academic subjects rather than product or service delivery.

In Project MiCRO a science teacher may pose a problem to his or her class such as how much rainfall is "ideal" for their locale. The students will create a computer data base showing optimum rainfall for such things as plant growth, drinking water supply, highway maintenance, and building construction. Then they will use spreadsheet software to determine the interrelationship of more or less rainfall on those items given their relative importance to each other.

In both examples a microcomputer is being used as a tool. The locus of control rests with the student. Also students are learning that technology can and should be used to aid critical thinking.

Critical Thinking Skills

In Project MiCRO a random sample of students participate in a critical thinking skills course. The rationale for this course is based upon the following assumptions. First, critical thinking is learned behavior, and given appropriate instruction almost any student can improve his or her skills in this area. Several psychologists, educators, and philosophers provide support for this assumption including Glaser, Sternberg, de Bono, and Lipman. Second, critical thinking skills should be taught directly, (Beyer). Third, the only way in which critical thinking skills are effectively transferred from one academic domain to another, for example computer programming to mathematics, is through the learner's mastery of those attributes which generalize to the thinking skills themselves (Taba).

Microcomputers are being used as the primary instructional delivery tool for several reasons. The excitement and motivation which microcomputers have caused among educators and students have created opportunities for rethinking curriculum and school organizational structure. Also, microcomputers permit the use of software to teach critical thinking skills in a highly individualized and interactive way. The ability of the teachers to simulate realistic, complex, sometimes dangerous, or historical situations in which to practice critical thinking is another advantage.

The objective of Project MiCRO which focuses upon teaching critical thinking skills is implemented with a random sample of 75-150 students in each school. One or two teachers in each school teach this course three days per week. It is designed to use interdisciplinary activities to teach specific thinking skills.

Project MiCRO uses reasonable, reflective thinking that is focused on deciding what to do or believe (Ennis), as a working definition of critical thinking.

Two levels of critical thinking are taught in Project MiCRO, discrete skills and multi-step processes. The combination of these skills and processes is what comprises critical thinking. The skills are: observation, recall, sequencing, classification, inference, induction, deduction, and association.

The processes selected for Project MiCRO are: distinguishing between fact and opinion, judging credibility, detecting bias, judging the relevance of information, solving problems, and making decisions. These processes involve a set of multi-step procedures which involve combining the skills listed above with other elements.

A modified version of the Pascal language with an interpreter, turtle graphics and animated figures is included in some activities, where appropriate, to teach specific thinking skills and processes. Simulation, logic courseware and application software are also used.

Teachers are trained to implement critical thinking skills using pedagogy that encourages intelligent behavior and metacognition among their students (Costa). Labeled examples and nonexamples of the skill, teacher modeling, guided discovery, open ended questioning, task-explicit positive reinforcement, adequate response time, a climate for divergent thinking and various points of view, facilitation of student interaction, and frequent assessment are some of the teaching behaviors stressed for teaching critical thinking.

Students also use peer-teaching. They work in small groups asking questions about each other's work that encourage critical thinking behaviors. Cooperation rather than competition is encouraged among students.

Teacher Training

An annual three to five day pre-service training session is conducted for all teachers. This training emphasizes integration of math, language arts, social studies, and science curriculum objectives into word processing, data base management, spreadsheet, and simulation activities. Trainers also work with instructional personnel so that they understand how to make courseware materials specifically relevant to the experiences of the students they teach, and the classroom management techniques needed to promote the greatest amount of time on task.

Pre-service training is followed by a concentrated period of in-service training. A trainer is on site regularly throughout the school year. The purpose of the site trainer is to provide additional training to teachers and principals, make classroom observations and demonstrations, assist teachers with courseware adaptations which make it relevant to the experiences of the children, and facilitate a good class management system for microcomputer use. The site trainer gives individualized attention to each teacher. She or he helps alleviate the fears of those who experience "computer anxiety".

A team of persons, including several teachers and the principal, has been identified by project and school staff to become a master team. They will gradually begin to assume responsibility for working with new teachers, monitoring the quality of the project, assessing staff development needs, and providing inservice training. By the end of the project, the expertise for its implementation and quality control will be institutionalized on site.

Parental Involvement

Parent coordinators, hired through Project MiCRO have contacted almost every parent in each of the three schools to familiarize them with Project MiCRO, and welcome them as active participants in the program. They each hold open house at the beginning of the school year so that parents can tour the computer laboratory.

Taking a microcomputer, the parent coordinators make home visits to introduce parents to Project MiCRO. Outreach enables the program to involve more parents than would otherwise occur, and helps to dispel the anxieties of those who are uncomfortable with technology.

Each school has a room designated as a parent room where parents frequently come to learn how to use microcomputers, and practice their skills. The parent coordinators meet with parents individually and in small groups to teach them to load and run software. Parents work on keyboard skills. Some become familiar with instructional software they can use at home with their children. The parent coordinator is available in the parent room at times convenient to parents including late afternoons and some evenings.

Parent coordinators periodically schedule workshops for parents on critical thinking skills, word processing, data base management and spreadsheets. Parents also seek assistance from parent coordinators in selecting home computers and software.

At all sites the parent coordinators perform multiple functions which have assisted the project in establishing a good rapport with the school as well as the community. They often help teachers with problems they encounter in running software. They conduct after school activities for students ranging from a microcomputer chess club to a stock market simulation. They provide information to the project director about implementation and instructional concerns.

The involvement of parents in the computer laboratory is a source of motivation and pride for both students and parents. Positive interaction and collaboration concerning instructional activities through Project MiCRO has transformed the relationship which many parents have with the school. Before the project began most parents visited the school only to handle a child's discipline problem. Often their contacts with school were negative. However, positive contact with the parent coordinator and instructional computing has encouraged parents to become involved with all aspects of the instructional program. It is expected that Project MiCRO parents will continue to apply this empowerment to future home-school relations after the project is complete.

EVALUATION

The evaluation of this project will answer the following three questions: Can the project design be implemented effectively? Does the project increase performance in critical thinking for those receiving the critical thinking skills course? Does the project result in an understanding of instructional computing for minority and economically disadvantaged students which they otherwise would not have?

Both formative and summative evaluation are being conducted. A pretest, post test control group design is being utilized. In addition, qualitative measures including anecdotal notes, interview and questionnaire data are being used to provide a richer, more complete description of important ecological factors such as student-teacher, student-computer, and student-student interactions, classroom climate, and learning environment.

The three Project MiCRO schools have been matched with three appropriate control schools on the basis of their comparability in several important factors including socio-economic status, geographic location, standardized achievement test scores, student racial composition,and students' prior experience with microcomputers. The control schools are A.W. Snowden School in North Carolina, Prichard Middle School in Alabama and Sammye Coan Middle School in Atlanta.

Although this Project is only half completed at this time, there are important findings which have implications for those interested in increasing opportunities for the targeted population.

Students' Computer Experience

Prior to implementation of the project, a questionnaire was designed to measure the students prior experience with microcomputers.

The questionnaire data are collected at control and treatment schools. Findings indicate:

- minority students do not have microcomputers in their homes;
- minority students do not take microcomputer classes outside the school setting;
- minority students in middle schools have had limited exposure to microcomputers in elementary schools.

Teacher's Perceptions of Microcomputer Usage

Prior to implementation teachers were asked to select from 14 computer competencies they felt confident performing. There were only two competencies which 75% or more of the teachers at the three treatment schools felt confident performing -- turning the microcomputer on, and taking care of hardware and software to prevent damage. When asked about their attitudes toward microcomputers, the vast majority of these teachers felt it either definitely or highly necessary to be computer literate for themselves and their students.

Computer Literacy (Pre and Post Test Results)

A test of computer literacy was developed by Carol Edwards primarily to measure students' knowledge of the microcomputer as a tool. The test of 20 questions has a range of 0-25 with seven factors -- locus of control, computer uses, ethics, data base, word processing, spreadsheet, and computer occupations. Pretest scores indicated a very limited knowledge. The mean scores were:

Treatment Schools	$\bar{X}$	Control Schools	$\bar{X}$
Pantego	11.45	Snowden	11.13
Walden	9.08	Coan	8.32
Dunbar	8.65	Prichard	8.26

A post test of computer literacy was given to a group of graduating eighth graders in June 1986, post test, computer literacy scores showed overall gains. The mean scores were:

Treatment Schools	$\bar{X}$	Control Schools	$\bar{X}$
Pantego	12.19	Snowden	10.72
Walden	12.60	Coan	n/a
Dunbar	n/a	Prichard	n/a

A t-test for difference between pre and post test was performed on these groups of students. The results indicated a significant gain($p > .001$) at Walden School, (N=79).

Test of Critical Thinking (Pretest)

After a thorough review of many tests of critical thinking, the Ennis Weir Essay Test of Critical Thinking was selected because its operational definition of critical thinking closely matched the project's definition, that is, a person's ability to appraise an argument and to formulate a written response. The test is not highly correlated with standardized tests of reading or math, can be group administered, and can be adapted to minimize cultural bias.

Pretest results of students in the control and experimental groups, indicate that these students have had little experience in this area. Students' average total score was 3.95 out of a possible 29.

Students did particularly poorly on the items related to recognition of irrelevance and misuse of analogy. Other analyses indicated no difference by sex or race of the student. The first post testing of critical thinking skills will occur in 1987.

Consumer Satisfaction Questionnaire

Data were obtained concerning how teachers, parents, and students felt about the project on topics which included learning processes, employment, self-esteem, locus of control parental involvement. The questionnaire was given to eighth grade Walden and Pantego teachers, parents, and students who were exposed to the treatment from March 1985 - June 1986. This section summarizes results from Walden and Pantego, and notes cross school trends.

Generally Project MiCRO is perceived positively by students, teachers, and parents. All agree that the project has increased academic learning. Walden and Pantego teachers concur 100% that Project MiCRO has helped students learn, parents, 80.5%. Eighty-nine percent of students at both schools also reported that the microcomputer enhanced academic learning.

Students, parents and teachers also agreed that Project MiCRO motivated students, and was fun. Low achievers as well as high achievers were perceived as being motivated by computers.

Project MiCRO participants concur that they would recommend the project to others. All teachers and 75% of all parents at both schools reported that they would recommend Project MiCRO to others.

Other interesting information emerged from the questionnaires. First, students became more aware of jobs in the computer industry even though Project MiCRO does not directly provide career information. Not only did many students express an interest in computer careers, but also began to see such aspirations as realistic and achievable. This perception could be significant considering the population served by the project.

Second, the fact that Project MiCRO is interesting, challenging and enjoyable to teachers probably contributed to their general job satisfaction. It also demonstrates that introducing a major innovation into schools does not have to be disruptive and anxiety-causing.

Finally, students, parents, and teachers agreed that Project MiCRO's reinforcement of positive self-esteem and locus of control is one factor which accounts for students' positive attitude toward the use of microcomputers.

Because these data are limited, it would be premature to make definitive statements about project results. These first questionnaire data do indicate, however, a very good beginning.

Other Observations

The experiences of the first two years of Project MiCRO have led to several observations. First, sex differentiation in microcomputer usage occurs most ofter during noninstructional computer time, for example, before and after school. Specific procedures and periodic monitoring are necessary to insure that girls receive the same opportunities as boys. Assigning a specific number of slots for girls in computer clubs, and making sure that girls enter programming contests are examples of two procedures that work well. Most important is to make sure the adult who works with students during non-formal computer time is constantly aware of including and encouraging girls in all activities.

Second, the instructional program benefits from having the active involvement of parents. This involvement can take several forms including more discussion in the home between parents and students regarding the student's school experiences, or more school visits by parents. Students and parents both view this involvement as very positive.

Third, in order that all teachers can use microcomputers as an effective teaching tool they need a great deal of staff development, much more than the average twenty hours most school districts offer. The most effective kind of training is the hands-on style like that offered by Project MiCRO site trainers rather than isolated workshops or lectures.

There is no substitute for experience. The more opportunities teachers have to actually work on microcomputers outside the classroom setting the more confident they become in the classroom. There are several effective ways of motivating teachers to use microcomputers including making them available during planning periods, overnight and on week-ends, and encouraging teachers to computerize their paper work.

Finally, using application software, simulations and logic games in an instructionally effective way requires a great deal of active teaching as well as solid pedagogic and classroom management techniques. Contrary to a misconception held by many teachers, students must be taught when using microcomputers. Teachers cannot take a passive role, simply rely on the hardware and software to do the job for them. Active teaching is especially necessary for microcomputer based critical thinking skills instruction.

Bibliography

Beyer, Barry. "Improving Thinking Skills - Defining the Problem." Phi Delta Kappan, March 1984, pp. 486-488.

-----. "Improving Thinking Skills - Practical Approaches." *Phi Delta Kappan*, April 1984, pp. 556-560.

Becker, Henry J. *School Uses of Microcomputers: Reports from a National Survey*. Center for Social Organization of Schools, Johns Hopkins University. Baltimore, MD., Issues 1-6, 1983-84.

-----. *School Uses of Microcomputers: Reports from the 1985 National Survey*. Center for Social Organization of Schools, Johns Hopkins University. Baltimore, MD.,

-----. "Unpublished data." Center for Social Organization of Schools, Johns Hopkins University. Baltimore, MD., 1986.

Costa, Arthur L. "Teaching Behaviors That Enable Student Thinking." *Developing Minds*. Ed. Arthur Costa. Association for Supervision & Curriculum Development, Autumn 1985, pp. 125-138.

de Bono, Edward. "Critical Thinking is not Enough." *Educational Leadership*, September 1984, pp. 16-17.

Ennis, Robert. "Critical Thinking and the Curriculum." *National Forum, The Phi Kappa Phi Journal*, Winter 1985, pp. 28-31.

-----. "Goals for a Critical Thinking Curriculum." *Developing Minds*. Ed. Arthur Costa. ASCD, Autumn 1985, pp. 54-57.

Glaser, Edward. "The Watson-Glaser Critical Thinking Appraisal." Unpublished paper presented at the Third International Conference on Critical Thinking and Educational Reform, Somona State University, July 1985.

Hood, John, et al. *Microcomputers in Schools*, 1984-85, A Comprehensive Survey and Analysis. Westport, CT 1985.

Lipman, Matthew. "Philosophy for Children." *Developing Minds*. Ed. Arthur Costa. ASCD, Autumn 1985, pp. 212-214.

Sternberg, Robert J. "Critical Thinking Its Nature, Measurement and Improvement." Essays on The Intellect. Ed. Frances R. Link. ASCD, 1985.

Taba, Hilda, et al. Thinking in Elementary School Children in San Francisco. San Francisco State College, U.S.O.E. Cooperative Research Project, 1964.

Whimbey, Arthur. "Test Results From Teaching Thinking." Developing Minds. Ed. Arthur Costa. ASCD, Autumn 1985, pp. 269-271.

SPECIAL STUDENT - COMPUTER CONNECTION

Beverly Cunningham
Council Bluffs Community Schools
Council Bluffs, Iowa 51501

Lee Cunningham
Educational Computing Innovations
Treynor, Iowa 51575

Abstract

Appropriate computer use is a must for special education students. For years these students have struggled to learn in quasi-conventional ways. These students have been removed from the regular classroom where they failed only to be placed in an environment that is identical to or all to similiar to the one in which they failed. It is not unusual then that they repeat the process. We are still adapting the child to our adult educational environment. The computer makes it possible to completely revitalize the special education curriculum and give children the greatest opportunity to work up to their potential.

The computer holds the key to: 1) turning subjects perceived as dull into exciting learning experiences, 2) keeping the children actively achieving with little, if any, added stimulus necessary, 3) turning abstract concepts into more learnable concrete ones, 4) creating environments to improve social skills, 5) teaching problem solving skills, and 6) individualizing instruction.

We have long been denying special students the opportunity to learn in their preferred style, the medium was simply not available. Today we have the computer which provides us the medium to open up an exciting educational world to our students.

We educators who have experienced the values of educational computing must exert every effort to assure that computers, appropriate educational software, and teachers prepared to use them effectively are available to our special education students.

Special Student - Computer Connection

Various publications indicate that more special education students are having the opportunity to use computers. This is very encouraging, however, their teachers often lack the formal training or personal experience necessary to use the computer appropriately. The appropriate use of the computer is a powerful tool to help special populations work up to their expectancy and to lead well adjusted, productive lives.

The capabilities of the computer are developing and expanding rapidly, not only as a stand alone tool but in conjunction with other technologies such as voice synthesizers, laser disks, and modems. Educators are, at best, currently taking advantage of only a small portion of the potential of the computer. It is difficult to visualize the impact effectively used computers and related technology could have on education. We must emphasize the "could have" since education has traditionally made changes very slowly and is often overly cautious when it comes to new innovations.

Everyone has different strengths and weaknesses, likes and dislikes, which inevitably leads to preferred learning styles. Traditionally schools have chosen to use a language based curriculum. Special education students frequently have difficulty adjusting to this learning style and are doomed to failure and frustration. We have a language based society but where is it written that everyone must receive the bulk of their education through the written word? J.R. Forsdale, in the American Annals of the Deaf, stated, "We in the instructional materials field have a resource we have underutilized: the learner himself. Although his needs have long guided us in the content of our materials, his capabilities have frequently been largely ignored." (Forsdale 1981) Students who have difficulty with reading can learn many concepts effectively with little or no reading on the computer. Isn't it better to learn without reading than not to learn at all? It can often be a lot more fun, even if you're a good reader!

Complete individualization has long been a dream with little hope of reality until the advent of the desktop computer. The computer will allow far more individualization. As more teachers become trained and gain experience with the capabilities of the computer and more software developers meet the needs of education, true individualization can become a reality.

One of the major problems that teachers have faced for years has been that of motivating students. Only a small percentage of students are motivated in all curricular areas. It is an unusual teacher who has the capability to continually teach in ways that are exciting to students. Dr. Lee Salk, world-renowned child psychologist, stated, "Essentially, the computer provides children with an opportunity to master things and to say, I've done that. It gives them a feeling of achievement. They can see it and feel it." (Wollman 1985) Success is one of the greatest sources for motivation.

As Seymour Papert states in Mindstorms, "It is not a justification for school math to say that despite its intrinsic dullness, inventive children can find excitement and meaning in it." (Papert 1980) Math is not the only subject that many students find dull in the standard curriculum. The computer is proving that what have formerly been considered "dull" subjects can become exciting. Also, as students accumulate experience with the computer it seems to encourage them to complete their regular work. Most of us have used different types of so called "motivators" over the years and find the useful life of each is usually short and we are constantly looking for new ideas. The computer has only been in the classroom a few years. Only time will tell if it will continue to be the outstanding motivator it has been to date. It is likely to continue as children can make the computer do things they want to do, they control the environment and that is sufficient motivation. It has lasted much longer than any other method that is commonly known and shows more signs of strengthening than weakening.

Social skills are extremely important to handicapped individuals. The quality of their social skills often makes the difference between acceptance and rejection by society as a whole. Many of us work just as diligently on social skills as we do on academics since we know this is often a decisive factor in the student's future. The computer offers a natural means for learning social skills.

The advantage of many software programs will be enhanced when students work in pairs. As we all know, when we work with others we are more likely to be cooperative and communicative. Working together with flashcards or a pegboard is a totally different situation than working together on the computer. Few students find any great excitement working on flashcards or pegboard. The computer has a magnetic attraction and holding power for most children. It noticeably improves their willingness to help each other and share information as compared to similar noncomputer activities. It's worthwhile to be pleasant with your partner if the alternative is to lose your turn. The natural good feelings and self esteem that come from positive relationships will often generalize from the computer experience to the classroom, lunchroom, playground, and home.

We all engage in problem solving on a regular basis often with little, if any, conscious thought that we are actually solving a problem. I'm sure you will agree that superior students have greater problem solving skills. Most of us have learned these skills without any deliberate teaching. Children who have difficulty learning usually are poor problem solvers, some extremely poor. Educators have long overlooked this phase of their education. It is difficult to teach problem solving in traditional teaching styles. There are many fine computer software programs that do a very good job of helping children learn problem solving skills. Logo is an outstanding example. The variety of situations and decisions they make while using these programs will generalize, under the guidance of knowledgable educators, to life beyond the computer. Problem solving skills are more universally useful than facts. Problem solving skills can be used in any situation throughout the individual's lifetime.

The majority of special education students were first considered problems because of their behavior. These students were frustrated because they couldn't succeed in the regular class. It is natural for children to act out, withdraw, or become disruptive when they are unable to cope with the demands made upon them. Adults generally mask these actions in more socially acceptable ways. A pattern of misbehavior is often extremely difficult to break. There seems to be something magical about the computer that motivates children to be better behaved. Classroom observations support this view and children do seem to respect the computer and behave accordingly. They like what the computer will do for them and the computer is a self disciplinarian in that they willingly learn to spell right and conform in ways that will assure it performs.

Computer learning should be a part of the curriculum, the same as math and reading. What teacher would even consider using math as a reward for good behavior or eliminating math for punishment? Students should be assured of their computer time but, just like math, it needs to be understood that certain behaviors will be expected.

Dr. Sylvia Weir wrote, "We are only beginning to understand the extent to which the educationally disabled have been denied rights and opportunities." (Weir 1982) Dr. Weir, a medical doctor (internal medicine, surgery, and pediatrics), BS physiology and histology, who has long worked in artificial intelligence and with special needs children, has written many articles and several books. She often states these children have hidden talents they don't have the opportunity to use in the traditional language based curriculum. Many computer programs provide children the opportunities to use these talents.

The least restrictive environment, which is mandated by federal law, often puts students in the position of being physically placed in a class with their regular education peers. All too often this is a token placement where the special student experiences the physical experience of being with nonhandicapped peers but doesn't have the sense of belonging that is so important.

Computers can make mainstreaming more of a genuine experience. As mentioned earlier, many special children do have hidden talents they aren't given the opportunity to use in the traditional language based curriculum. These same talents are often not developed in the regular education students which makes it possible for special students to actually out perform some peers in their strong areas. This gives regular education students the opportunity to see that special students do have valuable talents and both can combine talents to make a more productive learning experience.

School dropouts are usually individuals who experienced frustration and failure during their school years. The majority had few academic successes, poor social skills, and, consequently, very low self esteem. If children can experience success through computer use doesn't it stand to reason that there will be far fewer students who want to escape from the confines of the school environment? Computers are expensive, but not nearly as expensive to our society over the long term as dropouts.

The benefits of computer use mentioned in this paper are far from a complete list but it does illustrate the point that computers often hold the key to student success that has been difficult to attain within the traditional curriculum.

Teachers often feel they are so busy it's inconceivable to add more to their workload. Actually, when computers are appropriately used by both teachers and students the teacher will have more time to spend in other needed areas. The initial entry into computer use will take additional time but that time will reap a rich reward, for both teacher and student. Papert wrote, "The computer is the Proteus of machines. Its essence is its universality, its power to simulate. Because it can take on a thousand forms and can serve a thousand functions, it can appeal to a thousand tastes." (Papert 1980)

Classroom computer management is beyond the scope of this paper, however, as you gain experience, see your needs and those of your students, a management system will become visible. This system will include scheduling, teacher utilities, cataloging of disks, records of student daily computer use, and other items that are important to you.

We, the educators who see and have experienced the value in computing and related technologies, must exert our professional efforts as to how they should be used even more effectively with students. We must do everything within our power to convince our peers, administrators, parents, and, yes, even the people on the street, that education must be using the latest available technologies and giving extensive input to the developers of educational materials. Many large companies of yesteryear are no longer in existence as they did not keep up with the technological developments in their fields. How might we apply this to the educational field?

Jose Armando Valente's doctoral thesis involved the development of a computer based learning environment for children physically handicapped by cerebral palsy. Valente wrote, "It is not the child who should be adapted to the environment; rather, it is the other way around. The instructor should attempt to build bridges between parts of the environment relevant to the child's growth. Logo provides mechanisms for creating these bridges -- adding to the environment special materials. The instructor can provide bridges from the child to the environment through careful intervention." (Valente 1983) This seems equally true for all special students.

References

Forsdale, J. R., Designing Interactive Materials for Cost Effective Learning, American Annals of the Deaf, 1981, p. 722-726.

Papert, Seymour, Mindstorms: Children, Computers, and Powerful Ideas, Basic Books, Inc. 1980.

Valente, Jose Armando, Creating a Computer-Based Learning Environment for Physically Handicapped Children, Massachusetts Institute of Technology, 1983.

Weir, Dr. Sylvia, Logo: An Approach to Educating Disabled Children, Byte Magazine, September 1982, p. 342-360.

Wollman, Jane, Interview, When Lee Salk Talks, Parents Listen, Family Computing, October 1985, p. 35-36.

HIGH SCHOOL COMPUTER SCIENCE

Kathleen Zurhellen
Diane McGrath

ABSTRACT: Robotics in the Classroom: A Sourcebook and Data base

Kathleen Zurhellen, Chairperson and Editor, NYSCATE Professional Development Committee, St. Gabriel School, 590 West 235th St., Riverdale, New York 10463

Robotics in the Classroom: A Sourcebook and Database covers elementary grades, junior and senior high grades, higher education, and inservice in the state of New York. The information in the book is the result of a survey conducted by NYSCATE, an organization of educators who deal with computers and technology. Among the topics covered are: types of robotic exposure (eight different modalities), whether the exposure is "hands-on," types of instructional materials used (nine categories), a directory of robotics education and training institutions in New York and Pennsylvania, robotics course outlines, robotics manufacturers, and resource lists of films, periodicals and books.

The data base is made up of respondents' names, schools, addresses, phone numbers, and the type of robotics exposure that is currently available to students, together with the logistics and materials used. A printout of the data basesis in the book. The book is presently being used by the New York State Department of Education and The Center for Learning Technologies. Editor Kathleen Zurhellen welcomes responses from NECC attendees and plans to include them in future editions.

ABSTRACT: The Transfer of Problem Solving Skills from Programming in a First and Second Language

Diane McGrath, University of Illinois at Urbana-Champaign, Champaign, Illinois

The research to be presented will be the findings of a year-long study of high school students in four programming classes. Two of these classes were BASIC programming classes (the students' first language), and the other two were Pascal (the second language). One class of each language received explicit training in problem solving and the other did not. Control subjects were nonprogramming students. Students were given pretests and posttests on problems in six categories: creativity, analogical reasoning, debugging, reusable procedures, parameterized procedures, and the ability to use the conditional rule.

The theoretical framework for this research has two important parts: (1) We have traditionally expected to see transfer from programming to other areas of problem solving through the medium of only one inadequately learned programming language. We hypothesize that subjects will transfer the problem solving skills learned in programming when they have learned more than one programming language because they will be able to induce a number of abstract schemata that may be used in new problem situations. (2) Recent work on far transfer suggests that when a subject is trained using widely different examples, transfer is better than when a subject is trained using single domain examples only. Therefore, we hypothesize that explicit training on problem solving skills and using examples from other domains besides programming will improve the students' general problem solving abilities.

A MODEL CAI PROGRAM FOR APPLIED MATHEMATICS

Mary Garrett, CDP,CCP,CSP
Shiawassee County Intermediate School District
Vocational Accounting and Data Processing
Corunna, Mi. 48817

In 1985, the Shiawassee County Intermediate School District was granted $17,500 to create a model program for teaching Mathematics applications primarily for the special needs students (emotionally impaired, visually handicapped, mentally impaired, low reading ability, or deaf) within the Building Maintenance Program (a vocational program to teach high school students the practical skills necessary to get employment as housekeepers, custodians, building maintenance personnel, and building and grounds maintenance personnel). This presentation will present the program developed and briefly address the areas of equipment selection and program design. The program can be used by any students who need to learn the basic arithmetic operations with or without the practical applications. The applications are designed for the students within the Shiawassee County Building Maintenance Program and would need to be modified for other situations.

PROBLEM

When Frank Gregory, the Building Maintenance Instructor for the Shiawassee County Intermediate School District, attempted to teach his students to do applications such as calculating the number of gallons of paint needed to paint an area; determining the cost of carpet needed to recover a room's floor, filling out a requisition for restocking the supply room, etc., he discovered that a significant number of the students could not do the applications because they lacked the necessary basic arithmetic skills.

SOLUTION ATTEMPTED

With the aid of the Data Processing Instructor, a grant from the State of Michigan to help with equipment and programming costs, and four data processing students: Deborah K. Barber, Diana Perry, Steve Barnett, and John Lauro, a program was designed to not only teach students basic arithmetic skills, but to provide them with practical activities in the area of applied mathematics within the building maintenance area. A management program was incorporated so that the teacher could use the program with a minimum amount of record keeping and still have the students working at their own pace. By having MANY possible activities after each main topic area (counting, adding, subtracting, muitiplying, and dividing), the class could be kept together in movement from topic to topic while still allowing for self paced work. (Faster students merely would do more of the applications!). Seven computers with a connecting network to a printer and twenty megabyte hard drive were purchased.

THE PROGRAM

EDUCATIONAL PHILOSOPHY

The program was designed to be used at the beginning of a school year to help special needs students develop not only the skills, but the attitudes they will need to be able to use basic mathematical skills in the work they will do. The program, with the aid of a teacher to explain the concepts and principles, helps the student not only to learn the basic facts of arithmetic, but also how to "work out" the correct answer when the student either forgets the basic facts or is not certain that he remembers them correctly. The program not only helps the student work the correct answers, but DEMANDS that the student be correct three times in a row during practice sessions. This helps the student learn that it is better to slow down and be certain that he is doing the work correctly than to race through his work to get done, only to have many answers that are not correct. Also, by having practical applications immediately following the lessons, the student is able to see that learning to use the basic skills in arithmetic will be useful to him. By having the faster students do more activities while the slower students are finishing the lessons, the students can see that the important thing in this class is to get the job done correctly. If this takes some people more time than others, that is not a problem--as long as everyone does what he is assigned to do as quickly as he can do it correctly. If the student learns this at the same time he is learning class procedures, safety procedures, proper tool handling, and basic job skills during the first three to four weeks of a new school year, the student should find that his whole year in this class will run smoother because he has a good "I CAN DO IT" attitude.

OBJECTIVE:

The student will write the correct answer to building maintenance problems involving the use of basic arithmetic skills of counting, and whole number adding, subtracting, multiplying, or dividing and in combinations of all five skills in eight out of ten problems in each area following the procedures given either orally or in writing by the student's instructor.

ENABLING OBJECTIVES (basic skills):

The student will type the correct answer into a computer program that guides the student in the acquisition, practice, and demonstration of basic arithmetic skills in the area of counting, whole number adding, subtracting, multiplying, and dividing, practicing each skill until he/she can answer eight out of ten problems correctly.

ENABLING OBJECTIVES(simulated applications):
The student will type the correct answer into a computer program that simulates basic building maintenance applications using basic arithmetic skills in the area of counting, whole number adding, subtracting, multiplying, and dividing, practicing each application until he/she can answer eight out of ten problems correctly.

MAJOR PREMISE OF THE INSTRUCTIONAL MATERIALS:
If a student can 1) count and use the principle of equals which shows counting will arrive at the same answer as addition (subtraction), 2) use the principle of place value to separate larger problems into smaller parts, and 3) use carrying (borrowing) to move from one smaller part of a problem to another, the student will be able to master the basic skills in arithmetic.

MINOR PREMISE OF THE INSTRUCTIONAL MATERIALS:
If a student has mastered the basic skills in arithmetic, he will be able to apply them, given suitable directions and practice, to applications in building maintenance.

MINOR PREMISE OF THE CHOICE OF MEDIUM:
Computers will aid in the motivating of students to master basic skills by: 1). being intrinsicly interesting, 2). using movement and color to demonstrate skills, concepts, and principles, 3). providing immediate reinforcement for appropriate responses demonstrating the acquisition of components of the skills being mastered. Computers will allow the student to learn at his/her own pace and allow the instructor to monitor each student's progress by periodically printing reports of his/her accomplishments. Computers will allow the instructor to assign different students or groups of students different application tasks without being overcome with the problems of grading by allowing master correction programs to correct the student's application answers.

BASIC INFORMATION FOR USING THIS PROGRAM:
1). IF THE STUDENT HAS ANY PROBLEMS USING THE PROGRAM, IT IS THE PROGRAM'S FAULT, NOT THE STUDENT'S!

2). The computer is a TOOL to help the student learn the basic skills in working with numbers. The TEACHER is responsible for modifying assignments made by the computer or altering sequences, and has sole responsibility for determining grades. All of the programs have built in routines to help the student if errors occur in the practice exercises. On the practice exercises, the student must have three problems in a row correct to move on to a quiz. This is done to stress accuracy when working with basic arithmetic. When new topics are introduced, many of the quizes have help routines. If the student does not get the teacher determined percentage on the quiz, the program will go through the explanation and practice one more time and then require the quiz to be attempted again. If errors the program can not handle occur, or if the student attempts to do an application before finishing a started lesson, an error message is printed for the instructor. If the machine does not have a printer, the student will get a device timeout error. Explain the error to the student and have him restart the lesson.

3). The applications included are designed for the Shiawassee County Intermediate School District's Building Maintenance Program and should be checked for appropriateness with any other school's program. (All applications are on disk as ACT(topic number)(subtopic number)(file number).ASC and should be accessible with any word processor or EDLIN).

4). The application answers can be recorded on answer sheets and entered into the computer using the CAIM125.BAS and checked with CAICHECK.BAS. The number of problems attempted, correct, and possible are printed while the attempted and correct are recorded in the student file. CAICHECK.BAS allows the teacher to indicate what subtopic the student will attempt next.

5). The program CAIMFP.BAS allows the teacher to set up, update, examine, and report from the student's files. For each topic and subtopic, the number of problems attempted, number of problems answered correctly on the first try, the number attempted and correct on the last session, and the number of sessions for the topic are stored in the master record file, and in a report file so progress from time to time can be reported. The teacher can set the number of problems for the student to do and the percentage the student must have correct on the first try to move on to the next topic.

CAIMFP.BAS: MASTER FILE MANAGEMENT PROGRAM:
This program is designed to create the original master student files and the CAIMATH topic file. It will display on the screen information in either the students file or the topic file. Printed reports from either file may be obtained from this program also. To run this program the instructor simply types LOAD "CAIMFP.BAS" and then types RUN and the main menu will appear on the screen. From this menu the instructor may take his or her choice as to what they wish to do.

```
DO YOU WISH TO :
  <1> ADD TO PRESENT FILE
  <2> UPDATE STUDENT INFORMATION
  <3> DISPLAY STUDENT INFORMATION
  <4> DISPLAY STUDENT INDEX INFORMATION
  <5> DISPLAY BASIC CONCEPTS INFORMATION
  <6> MAKE BACK UP OF STUDENT FILES
  <7> PRINT STUDENT INFORMATION (RESET TO ZERO)
  <8> PRINT BASIC CONCEPTS TOPIC INFORMATION
  <9> END
```

Option 1 -- Allows the instructor to create new student master files or to add student master files. Along with the student master file a topic file for topic 1 (Basic Concepts) and a sequential file containing student numbers and student names are also created. Before allowing the instructor to add a student, the program will first ask if any student files have been created. If any files have been created the instructor must answer yes so that the sequential index file may be read in first before the new student is added.

Option 2 -- Allows the instructor to change program sequencing and problem assignment (topic, subtopic, number of problems or the percentage correct). This option may be used by the instructor when he wishes to display the task menu to see the topics and subtopics menus and choose what he wants the student to do.

Warning --This instructor should not make any changes to the student's file if the student is working on a machine with any of the CAIMATH programs as the student's program controls the updating of the file and will overwrite any changes may by the instructor at this time. After the student is finished the instructor to update.

Option 3 --Displays to the screen all topics and subtopic information which the student has done or is currently working on. Information for this comes from the student master file.

Option 4 --Displays to screen the sequential student file containing student number and student names. This option is taken when the instructor is unsure of the student number or if a check in needed to make sure the student name and number belong together.

Option 5 -- Display to the screen the information in the Basic Concept topic file.

Option 6 -- Backs all student data files to a designated drive. It should be used before taking option 7 incase there is any need for previous records.

Option 7 -- To be taken only at the end of a marking period this option sends to the printer for printing all the topic and subtopic information on which the student has worked or is currently working. Information for this report comes from the student master file. After the report is printed the student master file is checked to see if the student is currently working on a topic. If he or she has a topic that has yet to be completed, the current topic information is moved into the first position of the new marking period and the rest of the information is zeroed out. If the student has no current topic unfinished, the file is simply zeroed out. Conformation is required before the file is reset to zero.

Option 8 -- This option like option 7 is to be taken at the end of a marking period. Unlike option 7, the instructor is given the choice as to whether the file is to zeroed out or a printed report made. The report prints out the information from the Basic Concepts file.

Option 9 -- This option is taken when the instructor wishes to end the program.

CAIMATH.BAS: MASTER MENU PROGRAM.

This program is the program should be set to automatically boot when the student puts in the disk and turns on the computer if possible. If seperate boot disks are used, be certain to load BASIC with S/512 to set the file buffer for the large data files being used, before loading and runing this program. It requests the student's number and checks to see if the name matches the number. If the student does not know his number, the computer will provide a list of names for him to look up his number. If the student does not want to work on a lesson, this program will allow the student to choose from an application menu. The program checks the student's individual record and determines which lesson should be worked on at the current time, verifies that the student wants to work on that lesson, and then calls the appropriate lesson. This program is often called by the other programs, monitoring progress and calling lessons. Before using this program, the students should be introduced to the computer, its component parts (keyboard, monitor, CPU, disk drives, and printer). The students should be told and shown how to take care of the diskettes, how to place them in the computer, and how to turn on the computer. The student should be told/shown how to adjust the video monitor to make the image most comfortable for him. If a network is used, its function should be explained to the student so that much of the mystery of using computers is eliminated.

The student MUST be warned that all written responses must be typed in CAPITAL LETTERS (press the caps lock key to set the keyboard in capitals) and that the Enter Key (arrow pointing down and left next to the] key) must be pressed before the computer can process the student's response. In addition, the student should be encouraged to use the number keys at the top of the keyboard since an accidental pressing of the num lock key will turn of the number keys in the number pad. Students who have trouble reading should be encouraged to try to follow the directions as best they can. Most of the instruction is "carried out" using movement and color to reinforce the written instructions. Therefore even poor readers may be able to follow the "written" instructions by merely watching what happens and attempting to copy the action. If you are working on the TANDY machines and the program appears to have "locked up", check to see if the student has accidently pressed the hold key (by pressing it once). If the student accidently presses the CTRL key and the BREAK key at the same time, you can turn the program back on by typing CONT and pressing enter. The screen will be "messed up" until the next clear statement, but the program can continue. If you want the student to have a printed copy of any of the screens, press shift and the Prtsc key. The results will print on the printer assigned to the student's computer. If the student does not have a printer, the program may stop, but can be turned back on by typing CONT and pressing enter.

Topic 1,1 BASIC CONCEPTS, COUNTING:

IF THE STUDENT HAS ANY PROBLEMS USING THE PROGRAM, IT IS THE **PROGRAM**'S FAULT, NOT THE STUDENT'S!

THE PURPOSE OF THIS SUBTOPIC IS: to show the student how to count with the aid of the computer and to teach any student who does not have the skill, the skill of counting.

The following applications require only the ability to count and to record the results. It is suggested that the teacher "tell" each student the assignment (and give appropriate recording or inventory forms) to more closely simulate what would happen in the "real world", but each teacher must assess the ability of each student to follow directions and adjust the assignment according to that ability.

SAMPLES OF ACTIVITIES FOR THIS TOPIC:
The following cards are application activities for Topic 1 Subtopic 1: Counting. The answers may be entered using the program for subtopic 25 and checked and recorded with the checking program.

Activity 111

I. The School Board has decided that the windows in your school should all be replaced with clear plastic. Your boss wants you to determine how many will need to be ordered.
1). How many windows are in the front of your school?
2). How many windows are in the back of the school?
3). How many windows are on the right of the school (from front facing street)? 4). How many windows are on the left of the school (from front facing street)?

Activity 112

Periodic inventory must be taken of the supplies and equipment used by the building maintenance department. Get an inventory sheet and count and record the number of each item on the list.

Activity 113

The head custodian lost the master key to all the rooms inside your school. All the locks to the rooms and storage rooms must be replaced (including the tool room). (45). How many locks must be ordered to replace all of them?

Activity 114

In an attempt to reduce the water bill, all the water closets are to be replaced with new ones that conserve water. (46) How many water closets need to be ordered (include the water closets in the main office and in the workshop)?

Topic 1,2 BASIC CONCEPTS, PLACE VALUE:
IF THE STUDENT HAS ANY PROBLEMS USING THE PROGRAM, IT IS THE **PROGRAM'S** FAULT, NOT THE STUDENT'S!

THE PURPOSE OF THIS SUBTOPIC IS:
to show the student how big numbers can be broken into their component parts whose value depends on the position within the complete number.
This topic is one of the corner stones of the program and should be reinforced by the teacher in group discussions. The student who reads poorly and/or does not know the keyboard very well will have difficulty with this section since many of the answers are long answers like HUNDRED THOUSANDS that require both entry and reading skills. However, since the words are on the screen for the student to copy, most students CAN do the section, even if their reading and keying skills are very weak.
The following activities are NOT building maintenance applications, but are suggested activities to be done by the students in class or discussion groups to reinforce the principle of place value.

SAMPLES OF ACTIVITIES FOR THIS TOPIC:
The following cards are application activities for Topic 1 Subtopic 2:
Place Value. This is a set of GROUP DISCUSSION ASTIVITIES not application problems. If possible, do these BEFORE the students from the group working on the program reaches this subtopic.

Activity 121

1). Which would you rather have, 2 dimes or 12 pennies? Why?
2). Which would you rather have twenty one pennies or two dimes and one penny? Why?
3). Which would you rather have eight dollar bills or one ten dollar bill? Why?
4). Which would you rather have forty three dollar bills or four ten dollar bills and three dollar bills? Why?
5). How many ten dollar bills does it take to get a hundred dollar bill?
6). Using real dimes and pennies (on a overhead projector if possible so the relative sizes are very similiar) have the students take turns trying to "trick" each other into taking less for a set of coins.

Activity 123

7). Using play money dollar, ten, hundred, and thousand dollar bills, have the students take turns trying to "trick" each other into taking less for a set bills.

For really interested classes: Activity 124
8). Ask the students which they would rather have 11 base 2 dollars or 8 base ten dolars? Show them how you have tricked them. Repeat for base five. Stress the fact that place value in a base is the key to the size of a number, not the size of the digits. THE PURPOSE OF THIS ACTIVITY IS TO "PLAY" WITH PLACE VALUE, not TO TEACH THE STUDENTS DIFFERENT BASES. **HAVE FUN!!**

Activity 125

9). Write a very, very big number 1 on the board or overhead. Write a very, very small number 8 on the board or overhead. Ask the students which is bigger. Vote on the answer. Whatevery they vote show them how the definition of big makes the other answer correct. Vote again. If they have changed their position, show them how the definition of big makes their first answer correct!! Show how place value is big in value, not big in physical size. Talk about the difference between the size of a number and the size of a numeral.

Topic 1,3 BASIC CONCEPTS, ONE DIGIT ADDITION
IF THE STUDENT HAS ANY PROBLEMS USING THE PROGRAM, IT IS THE **PROGRAM'S** FAULT, NOT THE STUDENT'S!
THE PURPOSE OF THIS SUBTOPIC IS: to show the student the relationship between counting and addition (i.e. if the student can not remember the addition facts, he can count to find the sum).
The topic of the principle of equality which is used to "tie" counting to the arithmetic

operation of addition is one of the corner stones of the program and should be reinforced by the teacher in group discussions. The student who reads poorly will have little trouble with the principle of equals since color is used to show the student what needs to be entered.

The following activities are NOT building maintenance applications, but are suggested activities to be done by the students in class or discussion groups to reinforce the principle of place value.

ACTIVITY:

Put some pennies on the overhead. Arrange them into two groups. Count the number in each group. Add the numbers in the two groups. Count the entire amount. Point out that the two answers are the same because counting the two groups together is equal to adding to numbers in the 2 separate groups.

Repeat with other groupings. Repeat for other objects in the room.

QUESTION FOR GROUP DISCUSSION:

If counting will always give you the same answer as adding, why should you bother to "remember" the addition facts? (point out that even though being correct is more important, in doing applications on the job, speed is also important and counting is much, much slower.)

Topic 1.4 & 5:BASIC CONCEPTS, MORE THAN ONE DIGIT ADDITION:

THE PURPOSE OF THESE SUBTOPICS IS: to show the student that adding of numbers with more than one digit is merely a matter of splitting the problem into several one digit problems and doing each separately. This concept is heavily based on the student's understanding of place value.

Topic 1.6 & 7 BASIC CONCEPTS, ADDITION WITH A CARRY

Topic 1.8: BASIC CONCEPTS, LARGE NUMBER ADDITION

The students are ready for addition applications after finishing this topic. Sample activities:

Activity 133

Get a copy of the supply/tool inventory and answer how many of each of these there are:

6). pliers?
7). strippers?
8). hand tools?
9). chisels?
10). hand saws?
11). trowels?
12). saw blades?
13). hammers?
14). power tools
15). drills
16). saws
17). knives
18). measuring devices

Activity 137

Use Paxton/Patterson Catalog 86 to answer these questions.

22). What will be the total cost of one box of 12 adjust temple safty glasses model 34-2828 and one box of 12 model 34-2894 and one box of 12 model 34-2832?

23). What will be the total cost of one shop-vac dustman model 42-2146, one dust-master vacuum model 43-0062 and one one shop-vac light-duty 1.5 hp wet & dry vac model 42-3428?

TOPICS 1.9 - 14: SUBTRACTION

The subtopics of subtraction closely follow the subtopics of addition except that borrowing is used instead of carrying. Activities and discussions for addition can easily be adapted to subtraction if the teacher feels that the students will benefit from such activities and discussions. The students are ready for subtraction applications after finishing this topic.

Sample activities:

Activity 195

After checking the Custodial Supply/Equipment Inventory Sheet it was found that certain items were not up to maximum inventory number. How many of each item will need to be reordered?

(5) Disinfectant - inventory maximum 42 cases
- number used 31 cases
(6) Squeegees - inventory maximum 8 cases
- number used 3 cases

Activity 199

The school custodian has decided to polish all the floors in the school. It takes 2 cases of floor stripper and 3 cases of floor finish to do the floors. There is a total of 17 cases of floor stripper and 23 cases of floor finish.

(9) How many cases of floor stripper will there be left when the job is done?

(10) How many floor finish?

Activity 1913

If the plastic trash liners are changed every time the waste baskets are emptied How many liner are left in each box if: (20) 58" liners - total number used is 12 and the total number in box is 25

(21) 36" liners - total number used is 23 and the total number in box is 45

(22) 24" liners - total number used is 26 and the total number in box is 33

Activity 1921

Get a copy of the supply/tool inventory and answer how many of each of these there are if the following were lost:

32). 3 pliers?
33). 1 stripper?
34). 2 saws
35). 3 hand tools?
36). 2 chisels?
37). 4 saw blades?
38). 2 hammers?

Activity 1925 Use Paxton/Patterson Catalog 86 to answer these questions.

46). What will be the difference in the cost of one woodworking tool locker complete with tools and one building trades tool locker complete with tools?

TOPICS 1.15 - 19: MULTIPLICATION:

The subtopics of multiplication closely follow the subtopics of addition except that repeated addition is used if the student forgets the basic facts instead of counting, and the distributive principle is used to "split" larger number single digit problems into smaller problems based on counting by fives. The multiplication section assumes that the student can add and will call the instructor if problems occur that seem to concern addition. The last subtopic teaches the student to use a calculator to do multiplication. The teacher should take the time to discuss with the students when, where, and why to use a calculator and when, where, and why to do a multiplication problem by hand.

Most of the applications for the multiplication section involve extended totals (if one item costs $3 how much do 5 items cost) or determining area and volume.

TOPICS 1,20 - 24: DIVISION

The subtopics of Division closely follow the subtopics of multiplication except that repeated subtraction is used if the student forgets the basic facts instead of addition. The distributive principle is used to "split" larger number single digit problems into smaller problems based on multiplying by five. The division section assumes that the student can multiply and subtract. The last subtopic teaches the student to use a calculator to do division. The teacher should take the time to discuss with the students when, where, and why to use a calculator and when, where, and why to do a division problem by hand.

Most of the applications for division involve determining the cost of single items given group costs (if paint brushes cost $15 for 5, how much does one brush cost) or determining materials needed (how much paint is needed to cover the wall with area 600 square feet if each gallon covers 200 square feet)?

PROBLEMS WITH THE SOLUTION

At the time of the writing of this paper, only preliminary testing has been completed. The instructor choose not to do any non-application learning activities and the testing was done in the middle of the school year, after student behavior patterns had been set. Students completing the program showed some improvement in the post test of basic skills, but no attitude improvement. The testing indicated that some of the emotionally impaired students became VERY frustrated with the parts of the program that required three repeats of perfect performance to move on. They also became angry when the program sent them back for review, not just a repeat of the problem subtopic when they "failed" a quiz. The program was modified to allow more instructor intervention when it was needed. Now the program only repeats one section. Going back several sections is done only when the instructor decides review is necessary. Both the visually impaired and the low reading ability students complained that there was too much reading, even though they were assured that it was not necessary to read all the information on the screen to do the work. Most of the students found that the applications on the computer were boring and made little transfer to the "real" world. Although there still are a few applications (in case some teachers would want them), the Shiawassee County Building Maintenance Program did not use them. Instead of a purchase order simulation, a blank purchase order form was placed on the word processor and the students made "real" orders for materials for jobs they were doing or for supplies that needed ordering (using a calculator for the addition and multiplication and merely correcting any typing or arithmetic errors they made with the normal word processing correction routines). Students appeared much more motivated by the "real" applications than by the computer applications. Likewise simulations for calculating areas and materials needed to cover areas were found boring while the "real" applications were more interesting.

RECOMMENDATIONS

If the program described in this paper seems to be something that you would find useful in your classroom, I suggest you take three of four hours to try it (or have one of your students try it for you) and see if the basic mathematics instruction covered would be of use to your students. If you decide to use it, be certain to find applications for your program. For example, have the students help the librarian or the custodians take inventory or take inventory in your classroom; have the students help correct simple multiple choice & true false tests for other teachers and calculate percent grades, class average, student ranks, etc; have the students help the coach determine individual performances, team performance, rival team performances, etc; ask the custodian and/or lunch personnel to let the students help them figure costs of materials for jobs, ask the principal to let the students help figure busing, book and material, teacher aid costs; have the students figure costs, inventory, prices for the school store. The more real the application and relevant to the student's experiences, the more likely that the student will learn from the activities. If your students have a wide range of ability in mathematics, try forming teams were the entire team is graded by the total performance of EVERY member. Help the team captain assign tasks based on the members' abilities so that each team member is important, but does work within his level.

If you encounter problems with the program itself, most problems can be solved by merely resetting the topic and subtopic for the student using the CAIMFP.BAS and restarting the student's activity. All the code is on the disk. Feel free to ask the computer programming classes in your school to change it for you or to change it yourself. If you would be willing to share your experiences with the program or the applications with me, send your comments to the address listed under author.

MATHEMATICS AND PROBLEM SOLVING: COMPUTER BASED STRATEGIES FOR INSTRUCTION

Bethany N. Roditi
Anne Meyer

CAST Resources, Inc.
76 Lafayette Street
Salem, MA 01970

ABSTRACT

Teachers of learning disabled students are now faced with the dilemma of how to select computer software in order to systematically integrate it into the mathematics curriculum. The Center for Applied Special Technology (CAST) is developing a resource for teachers that will facilitate the integration of software into the curriculum. The Software Toolkit will be a series of books, one in each subject area, containing a detailed scope and sequence (K to 12), listings of selected high quality software that can be used to teach a particular set of skills, computer based teaching techniques, and thorough reviews of each program.

The Mathematics section of The Software Toolkit is based upon a scope and sequence that includes goals and objectives in the following general areas: whole numbers; problem solving; data collection, organization, and analysis; measurement, rational number, geometry and algebra. These mathematics skills serve as the framework for examining several well designed computer programs and for selecting software for the mathematics curriculum. The Mathematics Toolkit is used as a reference for supplementing instruction with computer based activities. Specific skill areas are addressed; software programs are comparatively reviewed and matched to instructional objectives; and. computer based activities are described. Deciding on which software to present, when, how, and in what order is now evolving into a developmental sequence for the presentation of math and problem solving software.

INTRODUCTION

Teachers of learning disabled students are now faced with the dilemma of how to select computer software in order to systematically integrate it into the mathematics curriculum. The goals of the curriculum for the 80's have shifted instruction from rote computation and concept development to include problem solving instruction (National Council of Teachers of Mathematics, 1984;1985; Fennema, 1981). Software is now available that reinforces basic math skills and, at the same time, encourages the development of problem solving strategies. With computers, several areas of the math curriculum can now be addressed simultaneously, and our toolkit of teaching techniques can be expanded to include supplementary computer based activities.

The Center for Applied Special Technology (CAST) is developing a resource for teachers that will facilitate the integration of software into the curriculum. The Software Toolkit will be a series of books, one in each subject area, containing a detailed scope and sequence (K to 12), listings of selected high quality software that can be used to teach a particular set of skills, computer based teaching techniques, and thorough reviews of each program.

The Mathematics section of The Software Toolkit is based upon a scope and sequence that is in the process of being developed. This curriculum includes goals and objectives in the following general areas: whole numbers; problem solving; data collection, organization, and analysis; measurement, rational numbers, geometry, algebra, and other. Considering the teacher's use of the computer to supplement instruction and remediation, the goals and objectives of the curriculum often intersect.

For example, the program Puzzle Tanks reinforces basic number facts and computational skills and simultaneously encourages the use of problem solving strategies. By manipulating the liquids in these tanks, the students try to move a targeted amount of liquid into the truck. In order to solve these problems, the students may have to add, subtract, multiply, and/or divide. They also have to problem solve: plan ahead, sequence their steps, think logically and reflectively, and create and test their hypotheses.

Through the incorporation this type of problem solving software into the mathematics curriculum, the instructional areas of whole number computation and problem solving intersect. The purpose of this talk is to further explain how these overlapping areas of the mathematics curriculum serve as the framework for examining several well designed computer programs and for selecting software for instruction. Deciding on which software to present to students, when, and in what order is now evolving into a developmental sequence for the presentation of computer software that interfaces with the scope and sequence for teaching mathematics.

Numerations.

In order to develop an understanding of whole numbers and counting skills, children must understand quantitative relationships that are represented physically, pictorially, and symbolically (Piaget, 1952; Bruner, 1966; Davidson, 1983). Traditional activities appropriate for teaching numeration concepts and skills to children include: having the children use manipulatives to count sets; labeling sets of objects with a corresponding number; counting items in a set of objects; and counting items in a set represented by pictures on a worksheet. There are several highly motivational software programs that reinforce early number concepts, and they offer immediate feedback and exciting rewards in the process.

Software Recommendations: For recognizing and identifying numbers and understanding quantity, Stickeybear Numbers by Weekly Reader Family Software, has been a useful introductory program. It allows the student to choose a number on the keyboard, and then colorful graphics represent ng that number are produced on the screen. The teacher can also introduce the concept of "counting on" or "taking one away" by pressing the spacebar.

Counting.

Once the student can recognize and identify numbers, counting principles have to be established (e.g., developing a one-to-one correspondence and a stable order for counting (Gelman, 1982).) Appropriate traditional activities include counting objects, counting pictures, matching sets visually displayed on a worksheet, and counting out loud using objects and pictures.

Software Recommendations: *Number Farm* by DLM reinforces the transfer of understanding quantity from physical representations to symbolic ones. Both numbers and number words are depicted along with the matching number of pictures and accompanying sounds. A menu is presented that offers six different counting formats. Visual and auditory stimuli are presented along with highly motivating graphics, rewards, immediate feedback, and second chances.

Addition.

Teaching the concept of addition is typically accomplished by using a variety of concrete aids and "hands-on activities". Students learn to make the transition from using physical objects to using number to describe added quantities. Some children with learning problems are often disorganized in their manipulation of concrete aids, and they lose attention if the teacher is demonstrating their use. Computers can be very effective in reinforcing the addition concept by depicting "addition" with objects moving on the screen. The blinking cursor assists students in paying attention and in focusing on the critical aspects of the problem.

Software Recommendations. Appropriate software for reinforcing the concept of addition includes *Houghton Mifflin Computational Games*, *MECC Elementary Volume 7 - Spaceship*, and *Stickeybear Math*.

This last program allows the teacher to customize the exercises to the student's needs. The teacher can adjust the difficulty of problems, can select addition and/ or subtraction problems with one or two digit addends with or without regrouping, can vary the formats from missing addends to vertical or horizontal presentations, and can limit the number of attempts the student can make. The student's prescription can be documented and saved on disk.

It is also very effective to allow the students to use concrete aids along with these programs and to count out loud while adding the quantities.

Treasure Hunt of Basic Facts by Josten Systems can be effectively used with accompanying manipulatives because the program can be controlled for time. In the untimed mode the learning disabled students can take as long as they need to figure out their answer by using concrete aids. This program works with a Ufonic speech synthesizer that offers immediate verbal feedback and directions. This feature allows our students to work more independently.

Recall of Number Facts.

Learning disabled students are often inaccurate and not automatic in their retrieval of basic whole number facts (Garnett and Fleischner, 1983; Ackerman, 1986). Recall of basic number facts can be reinforced quite effectively with the computer, because computers can provide drill and practice in a variety of highly motivational, multisensory formats for students who are having difficulty with memorization.

Software Recommendations. Teachers may draw on a wide variety of math software that provides drill and practice in a fun and motivating way. The teacher's selection will largely depend on the match between the students individual needs and learning style amd the idiosyncratic features and formats of the programs.

Alien Addition, part of the DLM *Arcademics*, is an example of a fast-paced video game format that offers the student practice in *accurate* and *automatic* recall of the addition facts from 0-9. The program can be adjusted by the teacher for facts up to 3, 6, or 9, and the teacher can control the speed of the program. Results can be documented, and student's progress can be charted. This type of program has proven to be very effective with students who are bored and "turned off" to flash cards and worksheet drills. With this game format, they can practice their facts and even have a little fun. For some students who have attention and impulsivity problems, this program may be too overwhelming and stressful. For these students who need an uncluttered screen, untimed or slower formats, immediate feedback, and second chances, other alternatives are available

Math Blaster by Davidson is a good example of a program that offers drill and practice in whole and rational numbers in a simplified format. One math fact is presented clearly on the screen. Teachers can adjust the program for longer response times, and review activities and a game option are also included.

Research has shown that approximately ten minutes a day of drill and practice using the computer can improve accuracy and build speed in number fact retrieval (Mason, 1985).

Problem Solving.

Teachers of learning disabled students often want to provide a variety of problem solving formats where the students can apply the arithmetic concepts and computational skills they have been taught. Traditionally, word problems were the problem-solving application. The computer offers a unique supplement to this aspect of the curriculum. The computer can be used to simulate or to model a problem situation that is difficult or impossible to encounter in the classroom. The monitor becomes a window into this "make-believe' or artificially contrived world that is logical.

Software Recommendations:
Puzzle Tanks by Sunburst is an excellent example of using the computer as a simulator. The child enters this problem solving environment and he acts on this environment by using the keyboard. With the arrow keys, the student manipulates liquids in two tanks.

Here the computer is being used to create a simulation environment within which students can practice problem-solving skills (e.g., using logical systematic approaches, planning, organizing information, developing strategies, and thinking flexibly. Interacting in this environment, the students are applying whole number arithmetic skills and at the same time they are problem-solving at a higher level in a format other than word problems.

Algorithms.

Students are eventually faced with learning and remembering to "regroup" in addition and to "borrow" in subtraction. For instruction and for practice, teachers have typically drawn from practice exercises in traditional math texts and worksheets. Some learning disabled students especially need assistance in learning the sequence of steps involved in working through the steps of an algorithm to solving an arithmetic problem. Some of these students have difficulty remembering to regroup while they are simultaneously recalling math facts, adding, and writing the digits in the proper place on the page.

Software Recommendations:
Teachers can draw on a wide variety of formats to teach and to reinforce math skills at the algorithm level. These formats include software packages that cover many aspects of the traditional curriculum. Software includes: Math Sequences by Milliken, and Computational Games by Houghton Mifflin. This form of software is available from math textbook publishers who offer software that supplements their text series. We have found that each subprogram has to be evaluated independently.

Software is also available in the forms of interactive strategy games, authorable shell games, and step-by-step tutorials. Strategy games similar to MasterMath, part of Mathematics Activities Courseware by Houghton

Mifflin, extend the curriculum beyond rote computation to include problem solving.

Master Math is an interactive strategy game for addition and subtraction that reinforces basic math facts, algorithms, and regrouping concepts and rules. Students who reason logically and who are reflective in their problem-solving approach are most apt to win even though their basic math facts have not been automatized.

Robomath by Mindplay is an example of an excellent shell game. Students can input their homework problems and solve them on the computer, and teachers can also program the math problems, control the type of operation (multiplication or division) , the level of difficulty (with remainders and regrouping or without), and the game itself (Joystick, mouse, or keyboard). In our lab, this game has proven to be a highly motivational instructional and remedial tool. Other substitutes include Game Frame I and II by Houghton Mifflin where the teachers can program short drill and practice questions.

An example of a well designed tutorial is Division Skills by Milton Bradley. Problems are presented one at a time and very clearly on the screen. The step-by-step tutorial establishes a routine for the students, and immediate feedback and cues are provided. Pre and Post tests are part of the program, and a teacher management system is included.

Fractions.

For teaching fraction concepts and skills to students with learning problems, pictorial representations on worksheets (e.g., pies) and manipulatives (e.g., Fraction Bars by Davidson) have been used. The teacher is now able to supplement instruction with well designed software.

Software Recommendations:
Software that reinforces fraction concepts include: Fraction Bars by Bennett and Assoc., a developmentally sequenced set of programs that coincide with the fraction bar manipulatives; Fraction Clues, one of the Mathematics Activities Courseware programs, and Adventures in Fractions by MECC.

This last program is especially helpful for learning disabled students because the tutorial is presented simply, clearly, and in a step-by-step fashion. A drill and practice game is also included, Ransom, that reinforces the "greater than"-"less than" concept using fractions.

Decimals.

Decimal concepts are typically taught after fractions in the traditional scope and sequence. Research has shown that students often acquire arithmetic concepts informally (Ginsburg, 1983) prior to formal instruction, and that students, especially those with special needs, need to develop problem-solving skills and strategies specific to each academic area including mathematics (Wong, 1985; Stone & Michals, 1986). By using the computer as a simulator, the concept of decimals can now be presented informally, and in the process, problem-solving skills can be fostered. Incidental learning about other quantitative and spatial relationships is also possible.

Software Recommendations:
Providing a problem-solving simulation environment like Mortars (Public Domain software) is cognitively and educationally valuable. As I present Mortars, think about its potential value in teaching mathematics and in teaching problem-solving: What mathematics is the LD student learning? Components include: geometry, ratio and proportion, greater than less than, estimation, decimals, data collection, analysis, and organizational skills, and number sense. What is the cognitive value of incorporating this simulation into the mathematics curriculum?

Selective Attention.

Some learning disabled children cannot focus their attention on the critical features of presented tasks such as arithmetic problems. Subsequently, they have difficulty formulating and applying strategies (Meltzer, 1984;, Campione and Brown, 1977).

Strategies relevant to processing mathematics include: cognitive flexibility, pattern analysis, using systematic logical approaches, orienting toward salient details, and self-monitoring and self-revision (Meltzer, 1983).

Software Recommendations: Computer simulations like Mortars by Public Domain, Puzzle Tanks by Sunburst, Baffles by Conduit are excellent instructional tools to teach problem-solving within the mathematics curriculum. While creating and testing their hypotheses, students are rewarded for using systematic logical approaches like the scientific method. Organizational skills are encouraged because note-taking and developing tables assist them in detecting patterns and relationships within this environment. The students self-check and revise their conjectures by estimating and reasoning logically and reflectively. The students are able to analyze their inaccurate approximations, and they learn from their errors. As facilitators, teachers can ask key questions to guide their students' thinking and to encourage strategy development and cognitive flexibility.

In general, using the computer as a simulator in the mathematics curriculum challenges the way we think about learning and teaching mathematics to LD students. Simulations are an excellent supplementary means of presenting a problem to solve. Upon encountering difficulty, the student is motivated to acquire and to use the necessary tools for solving the problem.

Estimation.

Teaching estimation to students is another important goal of math education. Appropriate activities include: guessing the number of jelly beans in the jar; estimating time and distance given a word problem. Teaching and drilling "rounding" and estimating skills can be effectively achieved using the computer because of the interactive quality of the software and the immediate feedback displaying the students' responses relative to the target goal.

Software Recommendations: Estimation games include: Power Drill by Sunburst and Tug of War by Houghton Mifflin. These are strategy games that encourage the student to round off and to estimate. The students must reflect upon their previous guess and reason logically to reach the next closer approximation. These games may be played with the computer or in teams. Simulations like Mortars are also excellent for teaching the concept of decimals.

Word Problems.

Word problems are difficult for some learning disabled students. The students must read and analyze the text, translate the text into a mathematical problem on paper, and then select a series of mathematical operations to solve the problem, follow the algorithms, and assess whether the answer is plausible. Students often have trouble selecting the critical information and solving the problem presented in this format.

Software Recommendations: Several programs are currently available to teach word problems. They include: Word Math by Houghton Mifflin, Math Word Problems by Weekly Reader Family Software, and Homework Helper: Math Word Problems by Spinnaker Software. Sem-Calc by Sunburst was pilot tested in our Lab and proved to be too difficult for all our LD students.

Abstract Responses.

Another part of the math curriculum centers on learning how to detect patterns, spatial relationships, and abstract reasoning. Some learning disabled students have difficulty in reasoning nonverbally and analyzing and synthesizing spatial information. To balance the mathematics curriculum with instruction on both quantity and space, teachers have appropriately used concrete aids, such as geoboards, cuisenaire rods, tangrams, and attribute blocks. Computer software is also available to supplement instruction and remediation in this area.

Software Recommendations: Programs that focus on sequencing and revisualization include: *The Factory* and *The Pond* by Sunburst. Software that reinforces problem solving and abstract reasoning using geometric shapes includes: *Gertrude's Secrets* and *Gertrude's Puzzles*. *Mortars* could also be included in this category. *Safari Search* is one of our favorite programs that encourages the students to develop search patterns, to rely on salient information, and to visualize patterns, and to create, test, and revise hypotheses.

Data Analysis.

One instructional goal is to teach the students how to collect, to organize, and to analyze data. Projects that involve language arts, other disciplines, and math & problem-solving are an exciting way of acquiring new knowledge and skills that cross several academic areas. Learning disabled students often find it difficult to organize themselves, and to reflect and to analyze information within this context.

Software Recommendations: The computer has the capability of representing the same data in multiple forms. One of the simplest programs is *Easy Graph* by Grolier. This interactive program offers a step-by-step tutorial on tables, vertical and horizontal bar graphs, pictographs, and pie charts. Once the students understand the concepts of data collection and graphing, they may use a more sophisticated program, *Exploring Tables, Graphs, and Charts* by Xerox. One side of this disk focuses on reviewing concepts and skills of reading and interpreting information using tables, graphs, and charts. On the second side of the disk, the computer becomes a creative tool, a blank tablet that can be edited and printed in the various graphing formats. For more advanced students, spreadsheet programs (e.g., *Edu-calc* by Grolier and *Appleworks*) are available. In the course of writing our book, we will be researching this area further in order to select the highest quality software and to recommend how it can be effectively incorporated into the mathematics curriculum.

As part of *The Software Toolkit* project, the mathematics scope and sequence will continue to be redesigned to accomodate the overlapping goals and objectives now possible by integrating the computer into the curriculum. We will continue to review and to select the highest quality software that can be used to teach and to remediate specific mathematics concepts, computational skills, and problem solving strategies. Our reviews will continue to be based upon the observations of instructional specialists who are teaching learning disabled and regular education students. It is our hope that this resource will save the teachers time and frustration, and give them the knowledge and the skills to select software and to plan computer based activities that effectively supplement their mathematics curriculum.

REFERENCES

Ackerman, P., Anhalt, J., Dykman, R. Arithmetic automatization failure in children with attention and reading disorders: associations and sequela, Journal of Learning Disabilities, 1986, 19, (4), 222-231.

Bruner, J. Toward a theory of instruction. Cambridge, MA: Belknap Press, 1966.

Campione J.C., & Brown, A.L. Toward a theory of intelligence: Continuations from research with retarded children. Intelligence, 2, 1978, 279-304.

Davidson, P. Mathematics learning viewed from a neurobiological model for intellectual functioning. Published report: National Institute of Education, Washington, D.C., 1983.

Fennema, E. (Ed.), Mathematics Education Research: Implications for the 80's, Alexandria, Va. Association for Supervision and Curriculum Development, 1981.

Garnett, K. & Fleischner, J. Automatization and basic fact performance of normal and learning disabled children. Learning Disability Quarterly, 1983, 6, 223-230.

Gelman, R. Basic numerical abilities. In R. Sternberg (Ed.) Advances in the psychology of human intelligence (vol. 1). N.J.:Lawrence Erlbaum Associates, 1982.

Ginsburg, H. (Ed.) The Development of Mathematical Thinking, N.Y.: Academic Press, Inc., 1983.

Meltzer, L.J. Educational assessment. In M.D. Levine, W.B.Carey, A.L. Crocker, and R.T. Gross (Eds.): Developmental-Behavioral Pediatrics, W.B. Saunders Co., 1983, 1100-1116.

Meltzer, L.J. Cognitive assessment and the diagnosis of learning problems. In M.D. Levine and P. Satz (Eds.), Middle Childhood: Development and dysfunction. Baltimore: University Park Press, 1984, 131-152.

Meltzer,L.J. Surveys of problem-solving and educational skills, Cambridge, MA: Educational Publishing Services, 1986.

Meltzer, L., Fenton, T., and Solomon, B. Automatization and abstract problem-solving as predictors of achievement. Paper presented at 92nd Annual Convention of the American Psychological Association at Toronto, Ontario, Canada, August 1984.

Meltzer,L., Roditi, B., and Fenton,T. Cognitive and learning profiles of delinquent and learning disabled adolescents, Adolescence, 21 (83), 1986, pp.581-591.

National Council of Teachers in Mathematics, The impact of computing technology on school mathematics: report of an NCTM conference, Journal of Computers in Mathematics and Science Teaching. Winter 1984/85, 19-22.

Piaget, J. The child's conception of number. N.Y.: W.W. Norton and Co., Inc., 1952.

Stone, C.A. and Michals, D., Problem solving skills in learning disabled children. In S.J. Ceci, Ed., Handbook of cognitive, social, and neuropsycholgical aspects of learning disabilities. Hillsdale, N.J.: Lawrence Hillbaum Assoc., 1986.

Wong, B. Issues in cognitive behavioral interventions in academic skill areas. In K.R. Harris, B.L. Wong, B.Keough (Eds.), Behavioral interventions: state of the art, Journal of Abnormal Child Psychology, 2, 1985.

REFERENCE NOTES

Swinney, D. and Smith, E. *Cognitive issues in understanding language processes*, A conference/workshop supported by the Max Planck Institute for Psycholinguistics, The Center for Psychosocial Studies, and the National Science Foundation. June, 1982, 19-26.

COMPUTER BASED LEARNING ENVIRONMENTS

Gertrude W. Abramson, Chair
City University of New York
New York, NY

ABSTRACT

Contemporary educational literature is replete with suggestions for changing curriculum and teaching to meet the needs of the information age; classrooms continue to be configured as per industrial age models, despite the introduction of electronic hardware. This session's panelists represent a wide spectrum of curriculum areas and grades. Each will present and justify a physical layout for computer based learning. The guiding principles are that designs be based on experience, judgement and vision of the future, bearing in mind that the student, not the furniture, is of primary concern.

PANELISTS

Theodore Abramson
City University of New York
New York, NY

Julie A. Brand
City University of New York
New York, NY

Anne Galletta
Academy for Educational Development
New York, NY

Fred Goldberg
Manhattan Technical Assistance Center
New York, NY

Audrey Harrigan
LaGuardia Community College
City University of New York

Carol A. Kaplan
Rockland Community College
State University of New York

Juan S. Mendez
Stuyvesant High School
New York, NY

EVALUATION OF DISTRICT INSTRUCTIONAL COMPUTER PROGRAMS

Karen Billings, Chair
Logo Computer Systems Inc.
Boston, MA

ABSTRACT

This presentation will address the major aspects of evaluating district instructional computer programs. Panel members will review the need for and interest in the evaluation process, discuss the steps in gathering and using that information, and offer summaries of recent evaluation projects around the country.

The session is designed for educators who want to review and perhaps revise their computer-related instruction. This review can be done by teachers, building principals, computer coordinators and other district personnel. A review of a course or the entire program can suggest what goals and objectives should be changed, what type of teacher/staff training is needed, if the appropriate hardware, software and print materials are being used, and if students are learning what is expected.

Many instructional programs get evaluated because the community, school board, or funding agency requests information and/or a report. However, the evaluation can provide information that can be used in the next stage of planning. Evaluations can both provide information for use in program revision and information that can be conveyed to interested parties.

The session will focus on the following three perspectives, each gained from a panelist's experience with the evaluation process.

General Frameworks of Evaluation: How do you determine the focus of an evaluation project? How do you collect the information and then analyze it? How can this information be used to influence the recommendations?

The first panelist will review the variety of frameworks and the specific steps in conducting an evaluation. She will pay particular attention to a review of the program goals, i.e., comparing the observed goals to the intended goals.

One District's Exercise with Evaluation: What lessons were learned in a school district that evaluated their computer program? The second panelist will offer caveats and advice, based on her recent evaluation experiences. She will pay particular attention to the involvement level of different people, the use of technology to help gather the data, the design of the tools for data collection, and the writing of the final report.

Results of Recent Evaluation Projects: What school districts have evaluated their instructional computer program in the last two years? What did they try to find out and what did they learn?

The third panelist will review the results of a formative evaluation project in 24 schools around the country, conducted as part of her doctoral dissertation study. She will describe the process of looking at four components of the program: (1) goals and objectives; (2) staff development; (3) instructional materials; and (4) student information. She will offer suggestions to educators interested in carrying out their own evaluation projects.

If time permits, panelists will summarize additional evaluation projects and suggest resources that are available to interested educators. The information will also be listed on handouts.

PANELISTS

Betty Collis
University of Victoria
Victoria, B.C., Canada

Sheila Cory
Chapel Hill-Carrboro City Schools
Chapel Hill, NC

DATA BASES IN THE CLASSROOM: BENEFICIAL FOR ALL STUDENTS?

Don Rawitsch
Minnesota Educational Computing Consortium
St. Paul, MN

ABSTRACT

The 1965-75 period in social studies education is strongly associated with the "inquiry movement." Significant attempts were made at that time to convince educators that students would benefit most from investigating critical issues through a process of proposing and testing student developed hypotheses. This activity in the social studies field paralleled efforts in other subject areas to emphasize "process" rather than fact acquisition.

The inquiry method did not revolutionize the teaching of social studies. Two significant problems in implementing inquiry activities were finding social science data for student analysis and obtaining data analysis tools suitable for student use. With the advent of widespread computer use in classrooms, a tool has become available. Personal computers make possible classroom use of utilities, software programs that manipulate, analyze and report information entered by the student.

Social studies teachers have been strongly urged to use utilities, such as the data base, with two objectives in mind: First, within the social studies subject area, this computerized tool represents a way for students to engage in the inquiry method. Second, within the realm of educational technology, school districts are integrating computer literacy programs into the curriculum. Many such programs include among their objectives that students will learn to instruct the computer to solve problems. The use of a data base utility is one way to do this without requiring knowledge of computer programming. Social studies is seen as a logical subject area for such activity.

Several assumptions are implicit in the call for data base use in social studies:

1. Data base activities and the skills they teach are valuable for all students;
2. Students will be able to make better use of problem solving thinking strategies for having learned to use a data base;
3. Students will be more motivated to learn social studies content and concepts when using action oriented data base activities; and
4. Data base use is an important carryover skill for adult life, the development of which is best started during the school years.

However, very little research is available to confirm these assumptions.

This session will primarily review the findings of a Minnesota study (scheduled for completion in spring, 1987) on this topic. This study, involving computer data base activities for eighth-grade students, will provide data to test the following hypotheses:

1. Students will be able to complete data base analysis activities faster when using a computer instead of paper and pencil methods;
2. Students will be able to complete data base analysis activities more accurately when using a computer instead of paper and pencil methods;
3. Students will like data base analysis activities better when using a computer instead of paper and pencil methods; and
4. Student success with computer data base activities will be related to individual learning style; more specifically, students with a "structured" style will be more successful than those with an "exploratory" style.

DATA BASES AND SPREADSHEETS IN HISTORY AND SOCIAL STUDIES INQUIRY

Larry Hannah
California State University
Sacramento, CA

ABSTRACT

Data base management has become a popular tool for inquiry in history and social studies. But where is the spreadsheet? Does it have any applications in history and social studies classrooms?

This presentation will focus on:

- Strategies for introducing data base management and spreadsheets as inquiry tools in history and social studies classrooms;
- Strategies for promoting higher-order thinking through tool use;
- Selecting the proper tool for the job; and
- Ways to train students to avoid inappropriate conclusions when using data base management and spreadsheets for inquiry.

SPREADSHEETS IN THE MATHEMATICS CURRICULUM: 6-12

Tony Jongejan
Western Washington University
Bellingham, WA

ABSTRACT

The computer with appropriate software can be used as a tool for facilitating the learning, teaching, and doing of mathematics. Of course, software is the key. This software can range from a drill and practice program to help students learn to add fractions, to using a programming language so students can write a program in Pascal to add fractions. Different types of software offer various degrees of flexibility for the teacher, power for the mathematician, and ease of use for the student. They also affect students' and teachers' perspectives of the computer itself.

There is an increasingly important class of software that is a reasonable compromise among flexibility, power, and ease of use. This software provides (a) an environment for users to work in and (b) tools to tailor this environment to their needs. Examples of software in this class are spreadsheets, graphing packages, *Geometry Supposer*, and *LogoWriter*.

This session will focus on the spreadsheet environment, which is a grid with tools such as labels, values, formulas, and functions. Students, teachers, and mathematicians can use the spreadsheet as a work area (worksheet) for the exploration and/or solution of mathematical problems, or they can design templates, which provide a specialized work area. Examples of templates that would be advantageous to students include:

1. Magic Squares: Students use the problem solving strategy of guess and check to fill in the magic square;
2. Compound Interest: Students compare the effect of various interest rates or compound periods;
3. Buying a Car: Students explore the actual cost of buying and driving a car;
4. Survey: Students explore weighted factors affecting livability in various cities and see a statistical analysis of the results;
5. Geometric Series: Students look at the sequence of partial sums as n gets quite large;
6. Buffalo: Students explore the effects of various management options on the survival of a buffalo herd;
7. Tickets: Students explore patterns in a classical mixture problem;
8. Exponential Growth and Decay: A chance to look at how the different variables affect this very important type of equation; and
9. Menu: A problem solving activity involving the combination of menu items to equal a given total (including tax).

As one can see, spreadsheet templates can be designed by teachers and used by students to better understand mathematical concepts as well as to explore problem solving settings.

IMPACT OF COMPUTERS ON THE MATHEMATICS CURRICULUM

Judi Mathis
University of Oregon
Eugene, OR

ABSTRACT

"The school curriculum does not yet reflect the advent of the calculator and the now readily available microcomputer to handle complex and iterative manipulations. Problems that are actually quite simple but were unsuitable for use in school mathematics because of tedious and time-consuming calculations can now be included at relatively low levels of mathematics instruction." —Mary Grace Kantowski

Calculators and computers have had an immense impact on our society, yet these changes have only been partially realized within education. In mathematics, it is forcing an analysis of the curriculum. The goals remain the same. However, the overall curriculum is changing. Just as technology is forcing this change, it is also providing new avenues to achieve the goals.

Before the state and district curriculum specialists reanalyze and restructure the mathematics curriculum, the following questions need to be discussed, debated, and developed:

1. What is the role of the computer in the teaching of mathematics?
2. Many computer and mathematics educators advocate an inverted curriculum. What does this mean?
3. What kind of computer software will be available for use in teaching Mathematics?
4. What are the possible uses of computers in the classroom?
5. How do the above ideas relate to my educational situation?

This session will provide some comments to stimulate discussion of these questions.

JUNIOR/COMMUNITY COLLEGE

J. D. Thomas
Dale Pugsley
Margaret Ellis
Jacqualine Couch

ABSTRACT: Courseware Authoring in All Departments at Freed-Hardeman College

J. D. Thomas, Vice President for Computing and Planning and Provost, Freed-Hardeman College, Henderson, TN

One or more faculty members from all 15 academic departments at Freed-Hardeman College have been trained to use Digital Equipment Corporation's Courseware Authoring System in five-week programs offered in the summers of 1984, 1985, and 1986. Courseware modules are being assigned in all 40 majors as part of a plan to insure that all students have multiple computing experiences. Sixty percent of the college faculty have extended computer training.

Key developments to date include effective trainers, a sophisticated authoring system, color graphics terminals in a student laboratory and a private faculty laboratory, and modest stipends for participation. A computer applications specialist and computer and information science faculty provide technical assistance, but all programming is done by faculty from the various disciplines. Courseware authoring was the first computing experience for several teachers, but most have proceeded to other applications.

Other workshops have been offered or are scheduled on Basic programming, word processing, teachers' utilities, the Statistical Package for the Social Sciences (SPSS^{X}), and integrated microcomputer packages. A leading edge telecommunications installation in 1986 provides computer network connections (along with telephone and video jacks) in every office, classroom, and laboratory. Objectives include desktop computing and a color graphics terminal in each department to facilitate further courseware development.

ABSTRACT: Computers and the Small Media Center

Dale Pugsley, Supervisor, Media Services, Western Nevada Community College, 2201 W. Nye Lane, Carson City, NV 89701

Media centers in small colleges have a different set of challenges from the larger centers. The larger centers usually have specialists who can devote full-time effort to a particular area. In addition to the traditional specialists in graphics, photography, maintenance, etc., they are often acquiring computer specialists, and/or using graduate computer specialists to develop needed computer applications for the centers.

In contrast, the small center may only have a staff of one or two. These people must be generalists and must perform all functions of the media center. Any progress into high technology, including computers, must just be an additional task assumed by the staff.

This session will show that being small does not necessarily mean being technologically behind. Small centers can also effectively move into the high tech areas.

The presenter will discuss what it takes in effort, desire, and ability on the part of the media staff to make the media center a leader in many areas of high technology. He will discuss the types of computer related services provided by his center, such as desktop publishing, administrative support, and providing computers for check-out by students and staff. He will emphasize the approaches he used in making these computer services a major function of the center without lowering the quality or quantity of traditional media services. He will present ideas on how to accomplish much with a limited budget.

ABSTRACT: Instructional Computing in Washington's Community Colleges

Margaret "Kit" Ellis, Instructional Computing Coordinator, Washington Community College Computing Consortium, 3860 159th Avenue NE, Suite 150, Redmond, WA 98052

The Washington Community College Computing Consortium (WCCCC) provides administrative and instructional computing support to Washington's 27 community colleges and the State Board for Community College Education. While the orientation for a central source of administrative support grew out of a legislative requirement for consistent reporting from the colleges, the value of a central support system for instructional computing carried a lower priority, therefore the benefits, although numerous, were realized at a slower pace.

Three years ago, the position of Instructional Computing Coordinator was established at the WCCCC to provide a focal point for supporting instructional computing throughout the community college system. The coordinator is selected from the community college faculty experienced in one or more aspects of instructional computing to serve for a two-year period.

A description of the governance structure of the Computing Consortium will explain the complexities of providing computing support to 27 nearly autonomous colleges. The benefits of the statewide system for both administrative and instructional computing will be discussed, as will the specific responsibilities and activities of the Instructional Computing Coordinator.

ABSTRACT: Computers, Disabled Students, and the Community College: Adapted Computer Studies for Disabled Students

Jacqualine Couch, Grossmont College, 8800 Grossmont College Drive, El Cajon, CA 92020

Community colleges are faced with the impending challenge of the growing enrollment of severely disabled students. Grossmont College currently has more than 500 disabled students out of a total student population of approximately 15,000 students. In order to more successfully mainstream these students into the regular college classroom or into a vocational program, some two years ago the Computer Science Information Systems Department of Grossmont College formally created a program of computer studies for the disabled student population. The purpose of this program is to provide disabled students with a tool—the computer—to assist in their collegiate or vocational efforts. Since the program's inception, approximately 15 percent of the total disabled student population annually enroll in this specialized program.

The purpose of this presentation is to share information on:

1. The overall structure of the Adapted Computer Studies program;
2. The successful utilization and structure of volunteer instructional aides within the program;
3. The types of specialized hardware and software that are essential to the program;
4. Problems encountered in teaching a multi-semester program open to all types of disabilities in a single classroom environment, and discovered solutions;
5. The future goals of the program based on an analysis of past performance.

A COMPUTING METHODS COURSE: CONTENT, ORGANIZATION, AND ACTIVITIES

Lajeane G. Thomas, Ed.D.
Louisiana Tech University
Teacher Education
Ruston, LA 71272

ABSTRACT

During the last two years, the number of secondary schools offering computer literacy and computer science as regular courses in the school curriculum has greatly increased. The infusion of these courses into school programs has resulted in the establishment of computer science teacher certification in many states. (Reinhold & Corkett, 1985). As a part of this certification, some agencies aré requiring a computing methods course. Interest in describing such a course has become widespread and colleges and universities are responding to the need for a computing methods course in a variety of ways.

This paper describes the computing methods course taught for the past two years at Louisiana Tech University. The course serves a broad audience that encompasses not only teachers of computer literacy and computer science, but also classroom teachers who wish to extend their knowledge of computers as a teaching and learning tool. The approach to the course content focuses primarily on how teachers may apply computers during each step in the instructional sequence. Model programs that may be modified to develop custom computer activities are the means by which programming concepts are presented. The application and synthesis of pedagogical concepts with content skills occur when each student develops a group of teaching materials for use in their own classroom.

This course has been a successful one for combining educational methodology and technical content and has been regarded by students as useful in developing skills needed for teaching "with" computers as well as teaching "about" computers. This paper describes the course by focusing on the course content, organization, and procedures.

BACKGROUND

Many colleges and universities are asking the question, "What is a computing methods course?" The sudden surge of interest in teacher certification in computer science and computer literacy has placed a certain sense of urgency to finding the answer to this question.

The interest in Louisiana for defining a computer methods course is fueled by the State's new computer literacy and computer science requirements which stipulate that at least one course in the certification sequence must be from a College of Education. National interest in a computing methods course gained momentum when a task force of the Association for Computing Machinery's (ACM Education Board, 1985) Elementary and Secondary School Subcommittee for Teacher Certification and Training recommended a course entitled, Methods of Teaching Computer Science, as a part of the proposed curriculum for computer science teacher certification (ACM Education Board, 1985). Since the focus of the committee was on computer science certification, the course described was aimed strictly at teachers of high school computer science.

At this university, the audience for a computing methods course is much broader than is served by the ACM course content. Many universities cannot justify the establishment of separate courses to serve secondary computer science teachers, other secondary teachers, and elementary school teachers. Therefore, a computing methods course at many universities must be flexible enough to serve a wide range of academic audiences and allow each participant to develop teaching strategies and materials to fit his instructional needs. This paper describes the computing methods course in terms of general information, content, organization, and activities. The technique of using model programs in developing pedagogical as well as technical expertise is given special emphasis.

GENERAL INFORMATION

Like other methods courses in colleges of education, it is important that students have some knowledge base as a foundation for the acquisition and application of methodology. It is very difficult to teach students to employ effective methods of teaching computer-related skills if the student does not yet possess the skills and content upon which the instruction in methods is based. A prerequisite for this course should be an introductory computing course that familiarizes the student with the following: how a computer works; uses of computers in our world; societal impact of computers; software evaluation; computer terminology; and programming of text and graphics in a higher level programming language.

The methods course is most effective at the senior level or higher. A student at this level who is seeking computer science certification will have completed many of his programming courses and content area methods courses. The student has reviewed commercially produced software in the introductory course and in content area methods courses and is familiar with lesson planning, psychological principles of learning, questioning techniques, content of his chosen subject area of emphasis, and has performed observations in a classroom setting. All of these experiences contribute greatly to the ability of the student to recognize how to organize instruction and how the computer can fit into the sequence of classroom instruction. Classroom teachers who are returning to the university to attain certification are particularly adept at recognizing applications of the CAI software, ways to develop teaching materials with the utility programs, and how the model programs can be modified to enhance instruction in their classrooms. Students with no experience in a classroom setting take much longer to recognize these applications.

Ideally, a course such as this could be taught by a team of faculty members from education and computer science. However, it is rare that this can occur. Therefore, if both cannot participate, this course should be taught by an education faculty member who is knowledgeable about computer science, or a computer science faculty member who is knowledgeable about education methodology, procedures, and content. It is quite necessary that the faculty have skills in both education and computer science because the course content involves in depth application of both skill areas. Although the course is organized around teaching methodology, a firm foundation of computer science knowledge and skills is essential.

COURSE CONTENT

The content of this computing methods course (See Figure I.) lays a foundation for use of computers in teaching and learning by reviewing aspects of planning for teaching that are present in any instructional endeavor and relating those specific activities and procedures to the use and learning of computer science concepts. The course addresses the pedagogy needed for teaching "with" computers as well as that needed for teaching "about" computers. Emphasis on computers in the curriculum, information resources to enhance independent learning, and general learning principles provide a knowledge base to which computing skills and concepts can be related. The course content focuses on teaching methods that facilitate students' learning of computer concepts and using the computer to augment the overall instructional process.

Figure I.

CONTENT OUTLINE FOR A
COMPUTING METHODS COURSE

I. Foundations of C.S. Education
  A. Overview of computers in education
  B. Computers in the school curriculum
  C. Information resources
  D. Instructional management
  E. Principles of learning
II. Aids to Instruction
  A. Utility software
  B. Software applications
  C. Types of instructional aids
III. Teaching "With" Computers
  A. Sequence of Instruction
    1. Motivation
    2. Prerequisite vocabulary/concepts
    3. Presentation of content
    4. Reinforcement and application
    5. Evaluation and management
  B. Instructional approaches
    1. Educational uses
    2. Classroom applications
IV. Teaching "About" Computers
  A. Prerequisite skills
    1. Keyboarding
    2. Terminology
    3. Prior programming concepts
    4. Top-down program design
    5. TEMPO teaching methods for computer-related skills (Thomas & Ohme, 1984)
  B. Programming Concepts
    1. One- and two-dimensional arrays
    2. High resolution graphics
    3. Animation
    4. Sound
    5. String functions
    6. Error handling and test data
    7. Introduction to Logo
    8. Advance topics - files, sorting, searching, pointers, recursion
  C. Modification of program models

The outline in Figure I. suggests a body of knowledge and skills concerning the use of computers as a tool for instruction and as an object of instruction. The topics themselves are not particularly unique. In fact, the description of this course so far is very typical--typical audience; typical foundations, activities, and content. The qualities that distinguish a methods course from other computing courses is not the WHAT of the course, but the HOW of the course--HOW the course is organized; HOW the programming concepts are presented; HOW the learnings are synthesized.

THE "HOW" OF THE COURSE

Course Organization

The organization of the course is somewhat unique in that instead of presenting the content on the basis of which programming statement or technique is to be learned next, the topics are based on the instructional sequences as shown in Figure II.

Figure II.

SEQUENCE OF TOPICS BASED ON AN
INSTRUCTIONAL SEQUENCE

- Foundations for Computer Use in School Settings
- Software Applications and Activities
- Motivation for Learning
- Development of Prerequisite Skills
- Presentation of Content to be Learned
- Reinforcement and Application of Learnings
- Evaluation and Management of Learning

As is suggested in the sequence of topics in Figure II., the stage is set by giving pedagogical information concerning use of computers in schools, integration of computer based instruction into the curriculum, learning principles, resources, and instructional management. Then the students are required to become familiar with a number of application programs that supply the tools needed to generate aids for instruction. Once the foundation is laid and the tools for instruction are available, the course organization centers on the steps in an instructional sequence and how computers can best fit into that sequence of learning procedures.

Presentation of Programming Concepts

As each step in the instructional sequence is emphasized and related to the use of computers, model programs (Culp, 1985) are introduced and commercial software reviewed. The model programs that demonstrate use of computers for each step in the instructional sequence include within them programming statements or procedures from the content outline. The model programs are designed so that they may be modified for use in a variety of subject areas. The procedures for using the model programs is shown in Figure III. below.

Figure III.

PROCEDURE FOR USE OF MODEL PROGRAMS

- Discuss the step in the instructional sequence
- Demonstrate software that can be used to fulfill that step
- Run the model program to see its usefulness for fulfilling the particular step in the instructional sequence
- Present programming concept(s)
- Demonstrate use of programming concepts by tracing execution of the model program
- Outline the procedure for modifying the model to create an instructional program that can be used to support the mode of instruction discussed
- Require modification of the model program by the student
- Assign a task which requires the student to use the programming concepts in a program he writes

It has been discovered from use of this procedure over the last two years, that model programs provide one of the major strengths of this course. Another strength is the organization according to the instructional sequence. These two procedures or methods of course organization work hand-in-hand to provide inservice or preservice teachers with a familiar framework through which they can relate the unfamiliar programming concepts to their background of experience and classroom computing needs.

Synthesis of Pedagogy and Technical Skills

The third strength of this course is the culminating activity which ties together all of the content and methodology presented throughout the course. It is a final project for which the student is required to develop a set of learning activities to be used in the classroom that will help students learn some skill or concept. It must include six or more activities that are designed to teach the concept or skill. Of the six, at least four must be computer based. The students are expected to use utility programs to develop some of the teaching materials and to apply the methodology and programming concepts presented during the course of study. The learning activities must be contained in some way, and must include a complete lesson plan that uses all the steps in the instructional sequence.

The students leave the course with a complete set of learning activities that includes on-the-computer and off-the-computer work designed to teach some concept that they identified. This assignment is very effective in drawing together the methodology and programming aspects of the content to develop a product that the students recognize as being useful to them in their respective teaching environments.

SUMMARY AND EVALUATION

The unique qualities of this course--organization according to the instructional sequence; use of model programs to present programming concepts; and synthesis of content and methodology through a comprehensive project--work together to provide an excellent learning environment. In written course evaluations, students have noted these reasons for feeling successful in the course:

- Programming concepts are presented in an understandable context: The model program provides a use for the programming procedure that is related to the student's background of experience.

- Program models can be used to develop instructional computing materials for any subject area: The students regard the programming concepts within the model programs as relevant because they can see an immediate application for the programming concept in an instructional setting.

- Learning proceeds from easy to more difficult and from concrete to abstract: There is enough structure to insure that the student has success in understanding the programming; the more experience the student has with the programming procedure, the less structure needs to be provided by the instructor.

- Models provide programming structures at a high skill level: The model programs exhibit good structure, documentation, and a sophisticated level of programming. Examination, modification, and application of these models results in the imitation of these desirable attributes in programs that the students write themselves.

Some of the attributes that students develop and exhibit during the course include the following:

- ability to list and explain steps in the design of a program;
- ability to trace the execution of programs and predict output;
- ability to evaluate errors in programs and tell what data or steps are lacking and then produce the changes in the program that will result in the desired outcome;
- tendency to work longer at solving problems for which the solution is not readily apparent; and
- tendency to show pride in their products and strive for accuracy, and efficiency.

The content, organizational structure, teaching procedures, and assignments for this computing methods course were carefully chosen to allow the flexibility for serving a wide range of academic audiences and for allowing each participant to develop teaching strategies and materials to fit his instructional computing needs. Results of evaluations of the course indicate that it is very successful in achieving the instructional goals and is perceived as effective and useful by the participants. Many participants have, in fact, reported that when using the units of instruction prepared in the course that their students were more motivated to learn the content presented, spent more time on task, and actually made better grades.

REFERENCES

Association for Computing Machinery Education Board. (1985). Proposed curriculum for programs leading to teacher certification in computer science. Communications of the ACM, 28(3), 275.

Culp, G. H. (1985). Introducing instructional computing in computer literacy. Unpublished manuscript. University of Texas, Austin.

Reinhold, F. & Corkett, K. (1985). EL's fifth annual survey of the states. Electronic Learning, 5(2), 25-31.

Thomas, L. G. & Ohme, P. A. (1984). Setting the TEMPO for instructional computing courses: an organized set of teaching methods for computer-related skills. [Summary]. Proceedings: Third Annual Micro-Computer Conference. Oklahoma State University, Stillwater, 3, 66-68.

THE DEVELOPMENT AND PILOTING OF A METHODS COURSE FOR HIGH SCHOOL COMPUTER SCIENCE TEACHERS

Dr. Charles S. White
Assistant Professor of Education
George Mason University
Fairfax, Virginia 22030

Dr. David C. Rine
Professor and Chair of Computer Science
George Mason University
Fairfax, Virginia 22030

As interest in offering Advanced Placement Computer Science (APCS) and APCS-related courses at the high school level has increased, so has the recognized need to provide sufficient preparation for those who will teach those courses. Computer science professionals have advanced a number of recommendations for computer science (CS) teacher training, and for certification programs for CS teachers, including a course that focuses on methods of teaching the concepts, skills, theories and practices of computer science. This paper describes the development and piloting of such a methods course through a collaborative project involving the Computer Science and Education departments at George Mason University (GMU) and 32 teachers in Fairfax County, Virginia during the spring and summer of 1986.

INTRODUCTION: ORIGINS

Only over the last four decades has computer science matured into a distinct discipline and has taken its place (in the late 1960s) with other disciplines as a university-level field of study. Much energy during the last two decades has been applied to strengthening the undergraduate computer science curriculum (C3S, 1968; C3S, 1978). More recently, however, these university computer science schools and departments have turned their attention (and recommendations) to pre-college education. In so doing, computer science embarked on the same path blazed by older disciplines. For example, as history and the social sciences came to be recognized as discrete university disciplines toward the end of the nineteenth century, each engaged in successful efforts to extend the teaching and learning of their discipline into the pre-college curriculum (i.e., social studies education). The same might be said of the various sciences that are common offerings in American secondary schools.

The last decade, then, has seen increasing discussion of how the pre-college curriculum might better prepare students to pursue serious study of computer science when they enter college. This discussion has been propelled further by the growing accessibility of computing to elementary and secondary school students, and the general efforts to provide "computer literacy" instruction for these students. By the early 1980s, committees associated with the College Board were sculpting the contours of advanced placement courses in computer science. This effort culminated in the development and administration of the first Advanced Placement-Computer Science (APCS) examination in 1984.

As interest in offering APCS and APCS-related courses has increased, so has the recognized need to provide sufficient preparation for those who will teach those courses. Computer science professionals have advanced a number of recommendations for computer science teacher training, and for certification programs for computer science teachers (Poirot, Taylor & Powell, 1981; Taylor & Poirot, 1984). One constant that tends to reappear in these recommendations is a methods course; that is, a course focusing on methods of teaching the concepts, skills, theories and practices of computer science. Since such a course requires a link between content and pedagogy, a logical course development strategy dictates collaboration between department of computer science and education, with the participation of teachers who already teach high school computer science. In the spring and summer of 1986, these elements came together in a project involving these two departments at George Mason University (GMU) and teachers in Fairfax County, Virginia.

THE PROJECT PROPOSAL

The Commonwealth of Virginia has not established certification standards for teachers of computer science, nor has the State developed a computer science curriculum designed to meet its particular needs. In northern Virginia, school divisions are challenged to devise a computer science curriculum that recognizes the unique characteristics of the region, where the computer industry and information technology are highly developed. Moreover, while the curriculum documents cited above provide a useful starting point, a computer science curriculum on the cutting edge must incorporate more recent concepts related to future computer systems, communications, expert systems, computer aided design and information science.

In the fall of 1985 and spring of 1986, the authors collaborated with the Fairfax County Public Schools to design a proposal with a two-fold purpose: (1) to illustrate the components of a rigorous AP computer science curriculum through the development of model curriculum materials and (2) to provide training to teachers at various levels of expertise in the delivery of a model computer science curriculum. The planning team responsible for the proposal included: (1) Dr. David Rine, project director, co-founder and chief reader of the College Board Advanced Placement Computer Science Program and professor of computer science at George Mason University, (2) Dr. Charles S. White, Director of the Center for Interactive Educational Technology and assistant professor of computer education at George Mason University, (3) Jerry Berry, Fairfax County Program Development Specialist, and (4) Marvin Koontz, Director of the Office of Technology for Fairfax County Public Schools.

Funding for the project, totalling approximately $107,000, was obtained from in-kind support from the University and Fairfax County Public Schools, as well as from a grant by the Center for Innovative Technology (CIT) in Herndon, Virginia. The CIT was established by the State of Virginia to facilitate links among universities, industry and public schools in part to foster technology transfer. This funding provided support for the project's key components: (1) development during the spring of 1986 of a methods course for computer science teachers, (2) delivery of the course during the summer of 1986, including 3 hours of graduate university credit and stipends for 32 participating teachers, summer support for project faculty and staff, and related expenses, and (3) materials refinement and evaluation during the winter of 1986-87.

THE COURSE

A course titled "Introduction to Computers" was listed in the GMU catalog under both the Computer Science and the Education departments as CS 611/EDUC 600, allowing for 3 graduate credits in either computer science or education. The class met daily from 8:30 am to 3:00 pm for four weeks, using the computing facilities at a local high school.

Course Content

Two general components formed the basis for the course: a computer science component and an educational methods component.

Computer science component. This aspect of the course flowed from our conception of a strong APCS curriculum characterized by the following features. First, such a course employs well-chosen applications to provide the motivating context for student work. Second, disciplined approaches to problem solving are applied, through carefully constructed programming methods. Third, considerable focus is turned to the means of achieving problem solutions (algorithm development). Fourth, a variety of ways to represent different kinds of information (data structures) are employed. Fifth, a good vehicle for implementing solutions on a computer (programming language) is used. Sixth, an examination of computer systems (hardware and software) illuminates what makes solutions work. Finally, the effects of computer use and the infusion of technology into society (social impact) are examined.

Educational methods component. Given these fundamental features of an APCS curriculum, the educational methods component explored ways to implement these features in an instructionally-effective manner. This goal suggests a number of key issues or topics to address: (1) criteria for the selection of problems, (2) strategies for helping students analyze problems, (3) approaches for teaching algorithm development, (4) ways to use student's knowledge of Logo to teach computer science concepts via Pascal, (5) strategies for teaching about computer systems, and (6) strategies for discussing the social impact of technology.

These instructional issues were addressed in two general ways. First, presentations and group work highlighted general instructional strategies and their application to computer science content. Such general strategies were identified in the areas of concept analysis and development, skill development, and problem solving. Second, expert teachers developed lessons incorporating and extending these general strategies; these lessons were then taught to the other participants and evaluated for further refinement.

Course Organization

The experience of participating teachers ranged broadly, from a few who have been teaching APCS-related courses for a couple of years to those whose experience has been limited to teaching general computer literacy classes. Since most high school computer science teachers currently do not possess undergraduate computer science backgrounds, even the most experienced practitioners benefited from the "filling-in-the-gaps" opportunity this course afforded. Given the experience range, participants were divided into three, nearly equal-sized groups for sessions involving computer science content. Group I (novices) studied programming methodology, features of programming languages, applications of computing, computer systems, and social implications. Reviewing Group I's content, Group II (intermediates) moved along to cover data types and structures, algorithms and additional applications of computers. Group III (experts) filled in the gaps of their previous experience and moved to other topics beyond what is covered in the typical APCS curriculum. Instruction of the three groups was accomplished through rotating morning sessions involving presentations by Dr. Rine and lab work (programming tasks in Pascal) overseen by Dr. White, Mr. Berry and a graduate assistant from the University.

The afternoon sessions were given over to the educational methods component, with all three goups meeting together. Two important benefits resulted from bringing the groups together for this component: the novices benefited from techniques the experts had developed over several semesters of APCS experience and the experts were challenged to help explain complex computer science concepts to less experienced colleagues, thus encouraging reflection on devising effective explanations.

A great amount of flexibility was accorded to the expert group, given rather individual learning needs and the additional lesson-development task we had assigned to them. The results were very pleasing, and the educational methods component benefited greatly from the combination of general methods presentations and model lesson presentations. Nine topics were addressed in the model lessons, including "An Introduction to Sets in Pascal," "Program Design and Specification," "Recursion," "Comparison of Logo and Pascal," "Loop Invariance," "An Introduction to Program Efficiency and Big-O Analysis," "Hashing," "Karel the Robot as an Introduction to Modular Programming," and "An Introduction to PROLOG." These model lessons are currently being refined by project faculty, will be re-evaluated by the expert teacher group, and will eventually be made available in a format yet to be determined.

Course Evaluation

Teacher participants were asked to provide feedback on the quality of the course at the conclusion of the four weeks. Two instruments were used to collect evaluation data. The first was a university-developed Likert scale containing general statements concerning the quality of course content and instruction. The second, more open-ended instrument designed by project faculty allowed extended comment and recommendations by the participants. The results of both were quite positive.

The first scale contained 11 positively-worded statements calling for agreement or disagreement (with available responses from 1 = strongly disagree to 5 = strongly agree). Items included "This course is taught well," "Presentations are clear and understandable," "My instructor seems well prepared for class," "My instructor stimulated my thinking," and the like. Responses to these statements were uniformly positive, with item means ranging from 4.21 to 4.89. One item relating to instructor awareness of current student progress did highlight, however, one diffcult challenge in a course organized by experience levels ($\underline{\bar{X}} = 3.98$).

Responses to the second instrument revealed how participants in the three different groups felt about the considerable amount and intensity of work done in the course. Not surprisingly, many respondents noted the heavy load, yet the vast majority of these viewed the load as necessary to achieve the purposes of the course. Moreover, the novices, who were working with Pascal for the first time, were no more likely to express dismay at the work load than members of the other two groups.

Participant comments solicited in the second instrument also confirmed the strength of the educational methods component, linking general instructional strategies and computer science content, supported by demonstrations of model lessons. One recommendation that did appear occasionally concerned altering the choice of topics for model lessons. The expert group concentrated part of their efforts on developing lessons on topics that they had found particularly challenging to convey to students. Some of these topics tended to be fairly advanced. While the novice group benefited from these, they would have benefited more from model lessons demonstrating other, more introductory computer science concepts. Such an approach helps novices make a stronger link between what they have just learned to how they might teach it to their students.

SUMMARY

In developing and piloting a methods course for computer science teachers, we addressed recent recommendations for teacher training and certification that have been advanced in the computer science education literature. Moreover, we attempted to address some of the more recent concepts and developments in the computer science field. We have been guided by the conviction that a high school course in computer science is more than a Pascal programming course, and that the language is only a vehicle to understanding the fundamental concepts and methods of computer science. Our conception of a methods course for computer science teachers reflects this conviction as well.

The experience gleaned from this project will inform further course refinement and program development at George Mason University. The successful collaboration between the Education and Computer Science departments has led us to begin designing a Computer Science Education Track within our existing Masters of Education degree specialization in Instructional Applications of Microcomputers. Preliminary plans call for coursework in both departments, with CS 611/ EDUC 600 as the synthesizing methods course. Such a program of study addresses a broader set of educational computing purposes.

Advanced Placement Computer Science is already having a major impact on undergraduate computer science programs of leading colleges and universities (Brennan, 1986; Dale, 1986; Mills, Basili, Gannon & Hamlet, 1987; Soloway, 1986; Young, 1986), and planned improvements in teacher training for pre-college computer science education, such as those reported in this paper, will improve the secondary school/ university interface.

REFERENCES

Brennan, J. (1986). Advanced Placement Computer Science from the university perspective. Proceedings of the 1986 National Educational Computing Conference (NECC '86), San Diego, CA: International Council for Computers in Education (ICCE), p. 198.

C3S [Curriculum Committee on Computer Science]. (1968). Curriculum '68, recommendations for academic programs in computer science. Communications of the ACM, 11(3), 151-197.

C3S [Curriculum Committee on Computer Science]. (1978). Curriculum '78, recommendations for the undergraduate program in computer science. Communications of the ACM, 22(3), 147-166.

Dale, N. (1986). Advanced Placement Computer Science from the high school perspective: Teachers speaking to teachers. Proceedings of the 1986 National Educational Computing Conference (NECC '86), San Diego, CA: ICCE, p. 217.

Mills, H., Basili, V., Gannon, J., & Hamlet, R. (1987). Teaching principles of computer programming. Proceedings of the Fifteenth Annual Computer Science Conference, Association for Computing Machinery, pp. 336-341.

Poirot, J., Taylor, R., & Powell, J. (1981), Teacher education. TOPICS: Computer Education for Elementary and Secondary Schools, Association for Computing Machinery, p. 18-27.

Soloway, E. (1986). Learning to program = learning to construct mechanisms and explanations. Communications of the ACM, 29(9), pp. 850-858.

Taylor, H. & Poirot, J.L. (1984). The certification of high school computer science teachers. Capitol-izing on Computers in Education, Proceedings of the 1984 Association of Educational Data Systems. Rockville, MD: Computer Science Press, pp. 295-305.

Young, F. (1986). Advanced placement in computer science: College level computer science in the high schools. Proceedings of the Fourteenth Annual Computer Science Conference. Association for Computing Machinery, p. 393.

LOGIC PROGRAMMING AS A COMPUTING ENVIRONMENT FOR DOCUMENTING TEACHER DECISION MAKING

Anne Shelly and Ernest Sibert

School of Computer and Information Science
Syracuse, New York 13244-1240

Abstract

Logic programming, a language for symbolic computation, is characterized as formalizing natural language and human reasoning (Kowalski, 1982). In general, our work in logic programming applications is directed toward aiding teachers and researchers in areas of education, the social sciences, and the humanities where computer-facilitated study has been relatively limited. These applications complement and facilitate text-based and/or induction-based methodologies in two ways: (a) they allow the user to work directly with symbolic information, and (b) they provide the means for mechanized deductive reasoning. We illustrate the capacities of such applications with examples from a system designed to facilitate and to explicate teacher decision making.

Conventional word-processing programs and LOGLISP, a system which integrates logic programming with LISP, form the basis for this set of programs. These programs aid the teacher's thinking in three areas: 1) creating, organizing, and storing a knowledge base of information about the students; 2) testing knowledge-base queries relevant to working hypotheses for confirming and for disconfirming knowledge-base information; and 3) creating, organizing, and storing files for the teacher's prose-text information and memoranda.

The present work grew out of an interest in the usefulness of applying symbolic computation and mechanized deduction to the analysis processes teachers use as they make instructional decisions. Their analyses involve observational and inductive processes as well as deductive processes. A complex effort is involved in the careful observation and interpretation of many sources of richly detailed information about students. Our interest is in applying computers to these activities, doing so in a manner consonant with the processes teachers use.

Teacher thinking is not a set of processes we wish to automate in its entirety. To do so would require that the computer understand human perceptions and interpretations at a very deep level -- one of the central problems of current Artificial Intelligence (AI) research. However, we see no need to automate the entire process of instructional decision making. Teachers are happy to carry out the interesting parts of the work, the induction, synthesis, and interpretation. We suggest that mechanical assistance has value if it facilitates parts of the process and if it explicates the process, thus contributing to our understanding of how teachers think.

We are exploring the use of an interactive system which functions as an intelligent assistant to the teacher. This system can understand (if only at a mechanical level) enough about teachers' analytic procedures to be convenient to use, and can take over not only elementary, though tedious, tasks, but also quite complex tasks which are nonetheless mechanizable using advanced computing methods.

We approached the development of this system using methods of computing which have evolved within the broad perspective of artificial intelligence research. The first of these emphasizes symbolic, as opposed to numeric, computation, and is best exemplified by LISP (McCarthy, 1960; McCarthy, Abrahams, Edwards, Hart, & Levin, 1965), one of the oldest and best known languages for symbolic computation. The symbolic approach offers the advantage that the user can express information using freely chosen symbols and creating expressions freely constructed from those symbols. The second computing methodology we have borrowed is that of mechanized deduction (Robinson, 1965), which makes it possible for the computer to deduce logical consequences of information provided by the user. In particular, we have used logic programming methods (Kowalski, 1974, 1979), which offer the advantage of very efficient implementation at the price of a slight restriction in expressive power. Using logic, users can enter information as assertions, which may be either particular statements of fact or general statements expressing rules. Such a collection of assertions is usually called a knowledge base. The implicit content of the knowledge base can be deduced in response to queries, which request the system to establish the existence of objects (actually, expressions denoting objects) satisfying some combination of logical conditions. Such an approach yields enormous flexibility, both for recording information, and for querying this information in ways that were not anticipated by the user at the time the information was entered.

The best known implementations of logic programming are the various forms of PROLOG (e.g., Roussel, 1975; Pereira, Pereira, & Warren, 1978). For these investigations, however, we have chosen a programming system, LOGLISP (Robinson & Sibert, 1982, 1984), which integrates logic with LISP and retains the benefits of both. In LOGLISP, information is represented and stored as *facts* (i.e., unconditional, specific statements asserted to be true) and as *rules* (i.e., possibly conditional, generalized statements asserted to be true). Users can develop *relations* (i.e., sets of assertions) by using words and symbols in forms they establish, thus creating representations of concepts they wish to express.

Recognizing that teachers' knowledge and perceptions of students and of the instructional process may vary, we emphasize the use of teacher-generated relations as a vehicle for representing information about students and about thinking processes that characterize each teacher's setting. We have established one such knowledge base in the system developed for student teachers who wish to practice instructional decision making (Morine-Dershimer, Harootunian, Shelly, Sibert, 1985). This knowledge base includes the following relations:

> Attendance, Classmember, Creativity, DataBank, Grades, Health, Parent, Parent-Conference, SeatChoice, Sibling, Sociogram, Stanford-W, Student-Interests, Teacher-Comment

Our choice of names for these relations is not constrained by the programming system; they are labels of our choosing, each representing a specific grouping of student information.

These relations differ not only in substance, but in structure as well. The most common form records specific information for each student within logic assertions. Using the convention that logic variables begin with a dollar sign, we can describe the form of the Attendance relation, for example, as:

```
Attendance $student $year $grade-level
           $present $absent $tardy
```

We can define the Attendance relation by assertions such as:

```
Attendance (Ira M) (81 82) 1 160 20 0
Attendance (Ira M) (82 83) 2 165 15 0
Attendance (Ira M) (83 84) 3 163 17 0
Attendance (Carmen V) (79 80) K 158 22 0
Attendance (Carmen V) (80 81) 1 160 20 0
Attemdance (Carmen V) (81 82) 1 164 16 0
Attendance (Carmen V) (82 83) 2 173 7 0
```

A second structuring device involves the use of text files. Lengthy, prose-text information for each student is contained within text files, and access information for the text relevant to each student is contained within logic assertions;

To illustrate, the Parent-Conference relation has the form:

```
Parent-Conference $text-file $file-lines
                  $student $group $fn
  where:

  $text-file and $file-lines denote
     text-file access information

  $student - denotes the student of reference

  $group - denotes the grade level
     for each instance of text

  $fn - denotes the academic year for each
     instance of text
```

and is defined by the assertion:

```
assert relation:
 Parent-Conference $file $lines $student $group $fn
if:
 Ftype $file ParConference
 Code $file $lines ^$student
 Fgroup $file $group
 Fnumber $file $fn
```

In this example, the relation is defined by the *predicate*, Parent-Conference, and by the five arguments, i.e., $file, $lines, $student, $group, $fn. A set of four conditions constrain the arguments by requiring them to meet specific relational characteristics. In the study discussed here, we decided to organize parent-conference notes in files labelled "ParConference". The *Ftype* condition provides access to these files. The files are grouped by grade level and are numbered by academic year. The *Fgroup* and *Fnumber* conditions provide access to these pieces of information. A portion of each file contains the conference notes for a particular student. The *Code* condition provides access to the portion(s) of the file previously labelled with the student's name by the researcher. Thus, the researcher can request parent-conference data in a variety of ways by mixing the combinations of specific information, e.g., using a specific student's name, and of logic variables, e.g., using $student.

We also use structures for relations in which information for all students is accessed through a combination of LISP functions and logic assertions, and in which information among relations is represented by higher-level relations.

Queries of the knowledge base can have one of two forms. In the first form, the teacher can search the knowledge base for all student information which matches the conditions s/he establishes for the search. To illustrate, let us assume the teacher wishes to review all parent-conference information on students who have had a history of high absenteeism. Using the student information in our knowledge base, the teacher might develop the following query:

```
>* search

answer template:
 >* $st absent $abs days in grade $gr has
        parent-conference $file $lines

conditions:
 >* Attendance $st $yr $gr $present $abs $tardy
 >* GE $abs 15
 >* Parent-Conference $file $lines $st $grp $fn
 >* end
```

In a "search", the teacher first identifies the form the answers should take: 1) using the symbol, "$", before words to signify logic variables for which answers will be supplied as results of running the query, and 2) using words or phrases that make the answer easier to read. Second, the teacher identifies the conditions of the "search" using a combination of teacher-generated relations (i.e., Attendance, Parent-Conference) and of relations built into the system (i.e., GE which stands for the "Greater than or Equal to" relation). A sample cf the answers returned by the program follows:

```
1 : (Bob W) absent 26 days in grade 2 has
       parent-conference KNDGTN-P80 (90 105)
2 : (Bob W) absent 26 days in grade 2 has
       parent-conference THIRD-P83 (352 363)
                        :
                        :
8 : (Ira M) absent 15 days in grade 2 has
       parent-conference KNDGTN-P80 (108 127)
9 : (Ira M) absent 15 days in grade 2 has
       parent-conference THIRD-P83 (428 460)
10 : (Ira M) absent 16 days in grade K has
        parent-conference KNDGTN-P80 (108 127)
11 : (Ira M) absent 16 days in grade K has
        parent-conference THIRD-P83 (428 460)
12 : (Ira M) absent 17 days in grade 3 has
        parent-conference KNDGTN-P80 (108 127)
14 : (Ira M) absent 19 days in grade 4 has
        parent-conference KNDGTN-P80 (108 127)
                        :
                        :
25 : (Jose D) absent 26 days in grade K has
        parent-conference THIRD-P83 (330 335)
26 : (June M) absent 15 days in grade 3 has
        parent-conference THIRD-P83 (151 217)
```

In the second type of query, the teacher might wish to explore a working hypothesis s/he has about the students. For example, the teacher might suggest the following hypothetical situation:

> When I group students for reading instruction, I look at standardized achievement subtest scores, comments by previous teachers, and previous report card grades. To be in my top reading group (in a fifth-grade classroom), the student should have: 1) above-grade-level scores on the reading and language achievement subtests, 2) teacher comments that reading performance is good, and 3) all A grades in reading.

Using our knowledge base, the teacher would form a hypothesis-test query of the form:

```
>* hytest

 answer template:
   $student

 if conditions:
  >* Stanford-W $student $yr $gr-lvl $age
       ($wdmean *) ($pp *) ($spell *) ($wdskill *)
       ($lang *) $mathcomp $mathconcept $scisoc
  >*Teacher-Comment $file * $student $group $fn
  >*Code $file $lines academic, rdg good
  >*Grades $student $year $gr-lvl Read $grade

 then conditions:
  >*GE $wdmean $gr-lvl
  >*GE $pp $gr-lvl
  >*GE $spell $gr-lvl
  >*GE $wdskill $gr-lvl
  >*= $grade A
  >*end
```

Using our knowledge base, the program responds with the following answers:

```
 6  confirming instances:
  1 : (Esther T)
  2 : (Jeff M)
  3 : (June M)
  4 : (Neil L)
  5 : (Robin E)
  6 : (Ruth B)

10  disconfirming instances:
  7 : (Bob W)
  8 : (Carmen V)
  9 : (Ira M)
 10 : (Jane A)
 11 : (Jim G)
 12 : (Joe K)
 13 : (John M)
 14 : (Jose D)
 15 : (Neil L)
 16 : (Robin E)
```

Confirming instances are cases which satisfy all of the "if conditions" and also all of the "then conditions." Disconfirming instances satisfy all of the "if conditions", but fail on one or more of the "then conditions."

These queries illustrate two points we wish to emphasize about system use:

1. Student information is easily accesible in ways that the teacher need not anticipate as s/he builds the knowledge base. Queries allow the teacher to efficiently call up the information s/he wishes by formalizing the characteristics that information should have. Thus the tedious tasks of rummaging through folders and records is eliminated.

2. Queries document the characteristics a teacher applies to any particular set of information. We know, from the "search" example, what this teacher means by the phrase "high absenteeism"; it means 15 or more days of absence in one year. From the "hytest" example, we can identify one possible operational definition of a top reading group. This type of documentation, combined with other chronological and topic information recorded and stored within the system, provides the teacher and/or researcher with a richly detailed data base of teacher thinking.

Figures 1 and 2 summarize the alternative task structures we suggest for system use. Figure 1 summarizes the system we are developing for use by student teachers wishing to practice their instructional decision-making skills (Morine-Dershimer, Harootunian, Shelly, Sibert, 1985). In figure 2, we suggest a similar type of system which allows the teacher to develop the knowledge base (i.e., instead of using an established knowledge base, as in the system for student teachers). As figure 2 suggests, the teacher builds his/her knowledge base, formulates questions about knowledge-base information relevant to his/her instructional focus, queries the knowledge base, adds additional information, reruns and/or reformulates queries, etc.

As the teacher proceeds, the system documents the analytic process in a variety of ways:

1. The knowledge base records the teacher's knowledge and perceptions as relations which can express specific constructs as well as connections among constructs.

2. Queries record the teacher's examination of the knowledge base to gather data or to test decisions around particular foci.

3. Memoranda record the teacher's rationales underlying knowledge base development and examination.

4. Chronological information automatically recorded in assertions and memoranda provides traces of the evolution of the teacher's thinking.

Ultimately, the teacher and/or researcher can analyze the data base captured by the system as a chronology of knowledge-base assertions and queries. The study of one such data base can produce a detailed case study of one teacher's thinking process; analysis of many such data bases can provide the basis for a grounded theory of teacher thinking (Glaser and Strauss, 1967). Thus, we suggest that such a system offers teachers and researchers the opportunity to explore teacher thinking and to increase the potential for more systematic study of those processes.

References

Glaser, B.G., & Strauss, A.L. (1967). The discovery of grounded theory: Strategies for qualitative research. New York: Aldine.

Kowalski, R.A. (1974). Predicate logic as programming language. In Proceedings IFIP 74 (pp. 569-574). Amsterdam: North-Holland.

Kowalski, R.A. (1979). Logic for problem solving. Amsterdam: North-Holland.

McCarthy, J. (1960). Recursive functions of symbolic expressions and their computation by machine, part I. Communications of the Association for Computing Machinery, 3, 184-195.

McCarthy, J., Abrahams, P.W., Edwards, D.J., Hart, T.P., & Levin, M.I. (1965). LISP 1.5 programmer's manual. 2nd ed., Cambridge, Mass.: The M.I.T. Press.

Morine-Dershimer, G., Harootunian, B., Shelly, A., & Sibert, E. (1985). Simulation on teacher decision making [Computer program]. Syracuse, New York: Syracuse University, School of Education and School of Computer and Information Science.

Pereira, L.M., Pereira, F., & Warren, D.H.D. (1978). User's guide to DEC-system10 PROLOG. (D.A.I. Research Paper No. 154). Edinburgh: Department of Artificial Intelligence, University of Edinburgh.

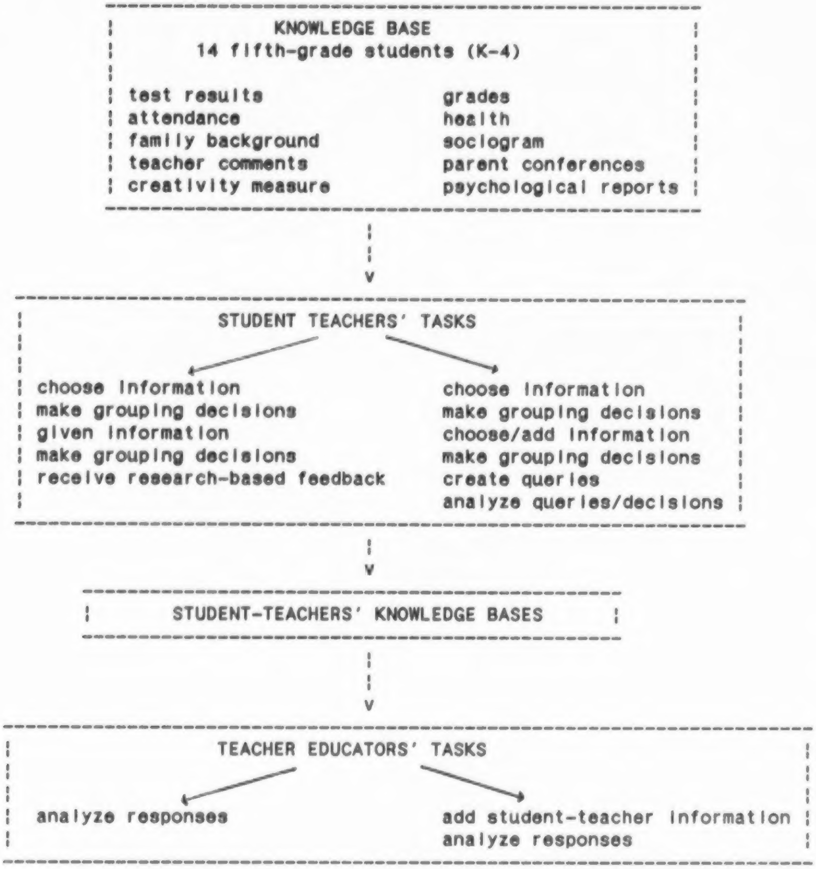
Robinson, J.A. (1965). A machine-oriented logic based on the resolution principle. Journal of the Association for Computing Machinery, 12, 23-41.

Robinson, J.A., & Sibert, E.E. (1982). LOGLISP: Motivation, design and implementation. In K.L. Clark & S.-A. Tarnlund (Eds.), Logic programming. (pp. 299-313). New York: Academic Press.

Robinson, J.A., & Sibert, E.E. (1984). The LOGLISP programming system. LPRC Technical Report. Syracuse, NY: Syracuse University, Logic Programming Research Center.

Roussel, P. (1975). PROLOG: manuel de référence et d'utilisation. Marseille: Groupe d'Intelligence Artificielle, U.E.R. de Luminy, Université d'Aix-Marseille II.

Figure 1
Simulation on Teacher Decision Making (STDM)*



KNOWLEDGE BASE
14 fifth-grade students (K-4)

test results	grades
attendance	health
family background	sociogram
teacher comments	parent conferences
creativity measure	psychological reports

STUDENT TEACHERS' TASKS

choose information	choose information
make grouping decisions	make grouping decisions
given information	choose/add information
make grouping decisions	make grouping decisions
receive research-based feedback	create queries
	analyze queries/decisions

STUDENT-TEACHERS' KNOWLEDGE BASES

TEACHER EDUCATORS' TASKS

analyze responses	add student-teacher information
	analyze responses

---

*Morine-Dershimer, G., Harootunian, B., Shelly, A., & Sibert, E.
Enhancing Teacher Decision Making Through Computer Simulations
Project Funded by the Fund for the Improvement of Postsecondary Education
(1985 -1987)

Figure 2
A LOGIC PROGRAMMING APPLICATION TO FACILITATE AND TO EXPLICATE
TEACHER DECISION MAKING

```
------------------------------------------------------------
¦                     KNOWLEDGE BASE                       ¦
¦      information on students stored as logic             ¦
¦            assertions and text; e.g.:                    ¦
¦                                                          ¦
¦ test results                   grades                    ¦
¦ attendance                     health                    ¦
¦ family background              sociogram                 ¦
¦ teacher comments               parent conferences        ¦
¦ creativity measure             psychological reports     ¦
------------------------------------------------------------


-------------------------------------------------------------------------
¦                        TEACHERS' PROCEDURES                           ¦
¦                                                                       ¦
¦                                                                       ¦
¦ build knowledge base                    build knowledge base          ¦
¦ create modeling queries                 create modeling queries       ¦
¦ expand/refine knowledge base            build research-based KB       ¦
¦ create new modeling queries             run queries using both KBs¦
¦ continue above cycle                    compare results of queries¦
¦                                         analyze queries/decisions     ¦
-------------------------------------------------------------------------
                                  ¦
                                  v
                ------------------------------------------------
                ¦  TEACHERS' METHODOLOGICAL KNOWLEDGE BASES    ¦
                ------------------------------------------------
                                  ¦
                                  ¦
                                  v
--------------------------------------------------------------------------------
¦                     TEACHERS' AND RESEARCHERS' TASKS                         ¦
¦                                                                              ¦
¦ add teacher information as KB assertions                                     ¦
¦ compare/contrast individual knowledge bases, i.e., teachers'  KBs            ¦
¦    and research-based KB                                                     ¦
¦ identify general model of teacher decision making using combined KBs         ¦
--------------------------------------------------------------------------------
```

EXPANDING COMPUTER LEARNING FOR DISADVANTAGED: QUALITY AND QUANTITY

Antonia Stone, Chair
Playing to Win
New York, NY

ABSTRACT

The word *equity*, like the word *literacy*, when paired with the word *computer*, means various things to various people--usually whatever they want it to mean.

Is computer equity equality of access? And if so, who is threatened? Who is being left out?

Is computer equity equality of instruction? And if so, who is threatened? Who is being left out?

Who are the "computer-disadvantaged"? Minorities? Women and girls? Children with special needs and handicaps? Urban kids? Rural kids? Adult nonreaders? Or just those who aren't special in any of the above ways?

Perhaps no one is against computer equity; perhaps no one actively wants segments of the population to be computer-disadvantaged. But this societal problem will not vanish simply because society at large passively ignores it. Programs that actively address computer inequities are required.

Panel members, each an activist with individual priorities, will address the issues raised above.

PANELISTS

Carol Edwards
Project Micro
Atlanta, GA

Stephanie Robinson
National Urban League
Washington, DC

Susan Jo Russell
Technical Education Resource Center (TERC)
Cambridge, MA

Richard Wenn
EPIE Institute
Kenwood, CA

SPONSOR: ICCE

BUGS IN THE WORKSHOP:
USING LOGO TO ENRICH LANGUAGE EXPERIENCE

Dr. Pamela Freyd
Coordinator of Computer Education
Graduate School of Education
University of Pennsylvania
3700 Walnut Street
Philadelphia, PA 19104
215-898-5690

Abstract

Logo can enhance language arts instruction because it offers language experiences beyond word processing and currently available language oriented software. The similarities between writing workshops and Logo environments and between writing processes and programming processes will be examined, along with Logo's potential for developing metalinguistic skills. The description of the Logo program in which these observations were made is included.

Introduction

This talk is about using the computer language Logo to enrich language arts instruction. It is based on three years of personal experience as a full time computer (Logo) specialist and from a theoretical background as a reading specialist and a classroom teacher. I hope to show how Logo fits naturally and easily into a curriculum in which reading and writing are viewed as communicative experiences rather than as sets of discrete skills. I will focus on three areas in which Logo can directly complement language instruction: a communicative classroom environment, a process approach to learning, and the development of metalinguistic skills.

Although now computer coordinator at the Graduate School of Education at the University of Pennsylvania, for the past 13 years I have taught at the Albert M. Greenfield School in downtown Philadelphia, an excitingly diversified K to 8 public school with wonderfully supportive parents.

In 1980, as an 8th grade English teacher working on my dissertation in the area of language acquisition, I used an 8K PET in a reaction time experiment delving into children's knowledge of derivational morphology. Although I finished and published the dissertation, I caught the computer bug in the process. In particular, I became intrigued by the contrast of my approach to the computer compared with children's. I came biased with cultural baggage with which children were not burdened. Boys and girls alike set out to explore and determine the limits of this new machine.

Two years later, when the parents purchased an Apple with 32 K, I asked to use it on Monday afternoons and found that I had the computer all the time since no one else in the school wanted it. There was no question about using software because a well-meaning parent broke our disk drive and half the year passed before its return. We did what we could, working and learning together in BASIC. As we progressed, however, I discovered that my students were doing on their own some of the things I had always wanted: they were assuming responsibility for their own learning on the computer; they organized access to our one computer; they worked together to solve problems. Independent, thoughtful, courteous classroom behavior seemed to be fostered, somehow, by the presence of the computer. I was intrigued. Was it just the novelty? The nature of the machine? The nature of the way we were using the machine? I returned once again to Penn, this time to study computers. Upon introduction to Logo, I knew I must try it with children.

In 1983, the Greenfield parents raised the funds to open a 32-microcomputer lab. The Commodore Company donated half of the computers. It was the desire of the principal, the parents and myself that this lab be used by all 920 children in the school, not just the mentally gifted who already had benefit of a school district computer program. We wanted students of all abilities, including the physically handicapped and the hearing impaired, to have an opportunity to learn to use the computer as a tool for learning. We believed that we could accomplish that best by initiating a Logo program to be supplemented with word processing. Our early planning was flexible so that we could let the program evolve through regular monitoring not only of our own progress but also of the progress of software development.

A computer lab arrangement with a specialist computer teacher is but one model of a way in which to introduce computers into a school. We considered use of the lab in this fashion to be transitional and hoped that after two years, computers would be used by all teachers as a tool for instruction. This did not happen, but that is a topic for another talk.

Although Logo is usually thought of in terms of its mathematical focus, we discovered that Logo seemed to have a potential to do things in the area of language arts beyond word processing and currently available language and reading software. I was particularly fortunate during the years that I worked in the computer lab, because students from the Computer Science Department at the University of Pennsylvania volunteered to help the children and to write Logo programs for us. We developed several programs hinting at the potential that Logo offers for exploration not only in math but also in language.

For the first two years all 920 children came to the lab in their classes. The 16 physically handicapped were mainstreamed, but I saw the hearing impaired as a separate class. It was a very heavy schedule. This year only 4th to 8th graders are scheduled to go to the lab in classes. The School District has provided seventeen computers to the school which means that computers can be put into the classrooms for the younger grades where they can be better integrated into the instructional curriculum.

Logo Environments

Because of its very nature, Logo provides an environment in which children are encouraged to discuss and solve problems, write and revise solutions, read and share results. This is a language oriented-environment, analogous to a workshop environment for writing.

The way in which computers are brought into and used in a classroom is a reflection of a teacher's and a system's educational philosophy. People inevitably project their philosophy of education onto computers in the way in which they use them. Traditional Computer Assisted Learning (CAI) has its roots in programmed learning, an approach that assumes that we give knowledge to children. The instructional material of CAI has been primarily in the form of tutorials and drill and practice exercises. These types of computer activities have been available for many years originating with large mainframe systems. Until very recently, however, these are the only types of programs have been available. Logo falls at the other end of our philosophical spectrum. Logo in any practical sense has only been available for the last few years even though its origins are significantly older. Logo is a higher level computer language based on the language LISP and Logo requires more computer memory than was available in microcomputers until recently.

Logo represents a constructionist approach to learning, discovery learning (Papert, 1980). A rule-governed microworld is created in which children can safely explore and discover principles for themselves. Some people feel threatened by the openness of Logo, in addition to feeling threatened by the very presence of the computer itself, because a great deal of classroom control must be relinquished. In a Logo environment, children ought to be exploring in different directions at different speeds.

Between the extremes, interesting material that takes advantage of the power of computers is beginning to appear. For example, many simulations are now available. The appearance of science probes and new approaches in social studies are providing opportunities for computer guided learning. Word processing and related prewriting and editing aids are of improving quality, tools that are most welcome by all of us.

Logo still represents the tool which with word processing most complements what we as reading/language teachers attempt to accomplish. Logo is a philosophy, a pedagogy, a computer language for children, but most important, it is characterized by an **open exploratory cooperative environment.** Three aspects of this environment are especially important for developing language skills in children.

Communicative Environment

By working with Logo, children can gain control of one part of their environment. Any type of programming puts the programmer in control of the computer. For a number of reasons, Logo is particularly well suited for children to learn to program. Although the main thrust is not to teach children to be programmers, being in control of a computer is a powerful feeling. Although we do not expect all children to become professional writers, we certainly hope to give them the power to express their feelings through written expression. We teach reading to empower children. For children who have experienced problems in learning to read, experiencing the successes which are virtually guaranteed with Logo can be a powerful motivation and a great support to self esteem. Logo is an empowering tool which may help children to feel in control of their environment and thus feel open to share their thoughts and feelings, to communicate.

Teachers also gain through knowing how to make Logo do things for them. One of the special characteristics of Logo is that it is a procedural language which means that a person can shape the structure of the language as he or she uses it. Most of us were introduced to Logo through turtle graphics. In that instance Logo was shaped to create an environment within which children can make discoveries about numbers and geometry. But Logo can also be shaped for discovery in other realms: a language discovery environment can be created. Programs can be written that encourage written language use as well as spoken use.

Our first effort at using Logo with language was to have children write short stories to accompany the pictures that they drew. We took advantage of the split screen. Younger children just added two or three lines at the keyboard and older children incorporated their text in programs. Students in the upper grades were given the challenge of writing and illustrating stories for children in the younger grades to read. We scheduled some double classes so that the older children could adopt the younger children and share stories because there wasn't enough of me to go around. Because we didn't have sufficient printer power for the large classes to make print-outs on any regular basis, we focused instead on moving stories and stories with music, activities that were a little different from usual paper writing and reading activities. In this case, the lack of printer capability really greatly increased the sharing and communication between students and provided real audience goals for the young writers. **DEMO** 5th grade story program.

We wrote two open-ended programs in Logo for our next venture into the language area. One, for the 4th to 8th graders, was an adaptation in Commodore Logo of Eliza, Joseph Weizenbaum's now classic program that simulates the behavior of a Rogerian psychotherapist. (*Computer Science Logo Style* by Brian Harvey, 1976, contains a fine annotated listing of this program.) DEMO This program generated hours and hours of writing on topics that the children selected. It produced unending requests for spelling help, even though the program accepts anything that is written. This program inspired many questions about how it worked and generated even more discussion about language use in general.

For the younger children we designed a simple program that let them type in words or sentences that seemed to make a circle person grow. We called this program Balloonman. **DEMO.** We began by asking the children to feed this person vegetables or fruits and then ventured into many variations. One group was so taken with Balloonman that they brought their dictionaries to class to help find the words they wanted. Most were content merely to discuss with their friends the spelling, content, and categories of words.

These two programs clearly increased the noise level of the already noisy lab, an indication of the amount of both oral and written communication taking place. But in addition, because the students were to some degree in control of Logo language, they could make inspections into how the programs worked. Although they were not sufficiently sophisticated to really understand all of a program such as Eliza, its inner workings were open to them. These programs motivated the children to use language and to ask questions about the nature of language and language use.

Cooperative Environments

Logo is a cooperative environment. Not only are children encouraged to work together, but teachers also work alongside children to help solve the programming problems that the children have set for themselves. There are numerous studies that support what all good teachers know, that children learn more when they learn in a cooperative environment (Johnson, Johnson & Stanne, 1986). When we teach writing, we try to develop a workshop atmosphere in the classroom, an atmosphere in which children are encouraged to help each other. In this atmosphere, children approach the teacher for help in what they are trying to express. A cooperative environment is a communicative environment.

Note the similarity of the comments from Papert and Graves in the following passages:

> *"In traditional schoolrooms, teachers do try to work collaboratively with children, but usually the material does not spontaneously generate research problems. Can an adult and a child genuinely collaborate on elementary school arithmetic? A very important feature of work with computers is that the teacher and the learner can be engaged in a real intellectual collaboration...Sharing the problem and the experience of solving it allows a child to learn from an adult not by 'doing what teacher says' but by 'doing what teacher does.*
>
> *Mindstorms 115*
> Seymour Papert

> *"We maintain their fiction (that writing is not a struggle) by not writing ourselves. Worse, we lose out on one of the most valuable ways to teach the craft. If they see us write, they will see us in the middle of the process, the hidden ground - from the choice of topic to the final completion of the work.*
>
> *Writing:Children and*
> *Teachers at Work 43*
> Donald Graves

A Logo environment is a communicative environment. In learning to program in Logo, children make decisions about what to make and how to communicate it. Because it is a cooperative environment, children talk to each other and help each other solve problems that arise as they try to make their programs work. Because Logo is process oriented, children begin to see making changes in their work as a natural part of creating something. Making changes, revising, is the essence of trying to be a better communicator. Not just good but a better communicator.

Process Approaches

Logo environments are concerned with processes rather than with products. Perhaps the most significant expression of this concern for process in Logo, is the fact that there is no such thing as a mistake or an error in the Logo language itself. There are times when the computer cannot understand what the Logo programmer is trying to do. There may be bugs, times or places when the Logo program doesn't do what the writer is trying to do. But the child is never told that he or she is wrong in Logo. This was terrifically difficult for my students to get used to. In fact, I think that it is fair to say that one of the biggest problems I had during my first year of teaching Logo was getting the children to let me see their programs that didn't do what they wanted. The children were so ingrained with the idea of 'wrong' that their approach to problem solving in Logo was to get rid of it if it didn't work.

Eventually, more and more children felt less and less threatened. Bit by bit the computer room became more process oriented. Slowly, rather than discard programs, we began to reexamine them together in order to try to find out just why they didn't do what was wanted. Since the projects were initiated by the children, I could never be the one to say whether or not it was right. Only the child could tell if the program was doing what he or she wanted. Ever so slowly, working style was changing. I was too busy to document this change, but I know that as requests for debugging help increased, the number of children that I could get to during any class period decreased. As in a writing workshop, the children increasingly helped each other.

The implications of this for the teaching of writing are significant, especially in the area of revision. Revision as a part of writing processes had been little studied before the appearance of computers and word processing because of the sheer physical effort of writing by hand. There were some who optimistically predicted that children would begin to revise once they were given word processing tools. Research by Daiute (1985) and others has shown, however, that children's approach to revision is at best local rather than global when one can get them to even consider revising. Children don't just naturally revise or even know how to go about it. Debugging approaches and attitudes developed in Logo programming can be applied to composition revision. Just bringing the notion of 'bug' into the writing workshop can introduce a non-threatening term into writing analysis. Logo processes support writing processes.

In addition, it is possible that Logo can help children better understand what revision is about in terms of matching goals with what has already been created. Children, even as old as eighth grade often fail to see how changes in one part of the composition affect other parts. They view revision locally. But if writing teachers know what is being encouraged in Logo classes, they can build upon the skills taught there. For example, children can see clearly how a change in one place in a Logo program can affect the whole picture. I mean this literally. The visual graphic domain supports this analysis. If this analogy is made clear to children, perhaps they can be led to understand the global consequences of revisions in their compositions, in the verbal domain. We are currently engaged in research examining this potential of Logo.

Programming and Writing Processes Compared

Let's look specifically now at some parallels between writing computer programs and writing compositions. In doing this, we may see ways in which skills gained in one domain may be brought to help in the other.

I am not going to try to convince you that Logo is better than other computer languages for developing programming style. Just let it be noted that when microcomputers first appeared, they had very little memory space. Programming style (or lack of style) developed to make maximum use of that space rather than with concern for legibility. Computer memory now, while not limitless, is sufficient so that saving memory space by the way a program is written is seldom a factor. As with the QWERTY typewriter keyboard, however, people who had invested time and thought in mastering the early systems such as BASIC became proponents of their continuation. That's why you may hear discussions about what programming language to teach children. Consensus is developing, however, that Logo is the best computer language for children, not only because it is easy to get started but also because it helps develop good style. Indeed, I might make even a stronger statement and claim that in the few years since its invention, Logo has become a classic component of American educational technology experience.

Randall English and Greg Edwards (1984) and Thomas Newkirk (1985) have noted the similarities between writing and programming. They point out that writing and programming involve the same major stages: prewriting and planning, composing, revising, editing, and publishing. Many of the same considerations much be taken into account. For example, both the writer and programmer must have in mind who the reader will be and what the background the of the person is. The programmer, however, can assess some of that information and let the program proceed from that assessment. Both the writer and programmer must decided what is the purpose, to teach or entertain. Both the writer and the programmer must decide on a focus. A writer can't cover American history in one page and a programmer can't cover all of mathematics in one lesson. Both writer and programmer must organize ideas. The writer may use topics and subtopics and organize them into some pattern. The programmer also must decide on organization. Both the writer and programmer must decide on a method. The writer might pick description, persuasion, comparison and contrast while the programmer might decide on whether to use a tutorial, drill and practice, simulations or games. Both the writer and programmer compose and both must revise and edit.

There are then many similarities between the processes of writing computer programs and composing. The most important one is that both seek to communicate ideas clearly and concisely. Writing teachers and Logo teachers are both concerned with developing style. DEMO - PARALLEL STRUCTURE. Writing teachers and Logo teachers are both concerned with helping children feel comfortable about making changes in their work to better meet the goals they have set for themselves. We feel that it is important that teachers talk about these processes with children and help them to see the parallel. We don't think that Logo offers any kind of 'easy fix' to inspire children to suddenly revise written work with adult skill. We do believe, however, that Logo processes enhance writing processes. Let Logo programming and composition writing be complementary activities.

Metalinguistic Skills

The last section of this talk focuses on Logo activities that may increase awareness of specific areas of language. I hope to show you ways in which Logo can foster metalinguistic development in children and also share with you some Logo ideas which support specific language skills. Logo's powerful language capabilities have been relatively neglected in articles and talks about Logo, probably because it's so very easy to get started working in Logo with graphics. But for those of us who teach reading and language, early exposure to Logo's list processing might actually be a more motivating introduction to Logo than graphics. Recent books provide much needed help in this area (Birch, 1986; Goldenberg & Feurzeig, 1987; Harvey, 1986; Lesser, 1985; Sharples, 1985).

Arbitrariness of Words

Bringing into focus the arbitrariness of words and their meanings is easy and fun in Logo. It is an aspect of language that even the little ones can appreciate with a Logo example. Recall that a Logo procedure can have any name; the procedure will behave according to the set of directions that are given to it. In fact, Logo *defines* procedures. Try writing the directions for a triangle into a procedure called square. Let is be the starting point for a discussion about words and their meanings. It can take the direction of noting the importance of good naming for procedures. Children are quick to see that the name of a procedure can be helpful in determining what it does. Point out the parallel of good names for titles or for topics in outlines.

The children in my classes loved to play with Logo names. The kindergarteners liked to have procedures called by their names so that they could practice typing their names to make the program run. Older children liked to name programs after their friends. While not necessarily always in the best programming style, this type of play was concerned with the nature of words and their meanings.

Cooperative Nature of Language

The fact that language between people is cooperative is another feature that computers (with help from teachers) can make explicit for children. For example, if a child misspells a word, a person will usually try to understand what is meant. Not so the precise binary coded computer. It doesn't know how to figure out what was intended. Working with programming is a wonderful way to help children begin to see important aspects of natural language. The computer language provides a measure for comparison. You might, for example, set up a demonstration in which you deliberately misspell the name and consequently get a response that there is not a procedure by that name. Or you might use an example when a situation arises naturally with students. In either case, take time from the class to call attention to the fact that people understand and use language in ways that are different from computers - at least at this stage in computer development. Let children begin to note those ways. You might keep a chart on the wall to which children could add their observations about language as they arise.

Word Classes

A traditional and playful technique for reinforcing children's experience with parts of speech is the use of Madlibs. DEMO. Interactive programming adds a new dimension to games that can be played with word classes. Using Logo to create Madlibs allows children to create their own easily. These can be written at a level of sophistication both in programming level and in language level to match the students (and teachers) ability.

Syntax

Programs that randomly generate word strings from lists of words can be used to help children become aware of the existence of different word classes and of ways that word classes fit together. DEMO. Note that the need to organize these word classes arises from the task that we have set for ourselves - to have the computer write sentences from lists of words.

Once these word classes are in place, students will begin to see the need for rules about such things as subject-verb agreement. DEMO. Rather than teach rules, let the children figure out the rules in order to make the program work. These are learning opportunities in language that didn't exist before computers.

A variation on this program might be to have students create form letters. With a teacher's guidance, children may be led to discover and analyze transitional phrases, common phrases for particular types of writing. This is a totally different way to approach language learning than with a grammar book.

Interactive programming can involve children in language tasks in which they will see for themselves reasons for knowing how to parse sentences, that is knowing what information can be gained from word order, for example. There are challenging programming tasks for junior and senior high-schoolers that encourage this type of deep thinking about the structure of language. Although I have not seen it at the time of this writing, *Exploring Language with Logo* by Goldenberg and Feurzeig ought to be an excellent source for such activities.

Self-monitoring

Finally, self-monitoring is an important metacognitive ability in any domain. Clements and Gullo (1984) report evidence that students who worked with Logo outperformed other students in 1) deciding on the nature of a problem 2) selecting a mental representation and 3) monitoring comprehension. The evidence is far from conclusive, but most certainly worth greater study. Whether or not strategies of self-monitoring in Logo might extend to reading and writing problems is yet to be determined. Perhaps it will turn out to be the case that certain working environments such as found in Logo and writing workshops promote greater self-monitoring. It is certainly an interesting question.

Conclusion

A Logo environment is a communicative environment and language teachers know that a communicative environment is important to language learning. In a communicative environment children are not afraid to try things and ask questions. They feel comfortable writing and talking. A Logo environment is a process oriented environment. Exciting research is just now taking place that extends our understanding of writing processes and reading processes (Bereiter & Scardamalia, 1986; Flower et al, 1985; Freedman, 1985). Logo has the potential to enhance writing processes. By learning Logo, children can gain opportunities to play with and explore the structure of language in ways that other computer activities do not offer. We hope that you begin to see why we feel that Logo can offer language arts experiences beyond word processing and currently available software.

Bibliography

Bereiter, G & Scardamalia, M. (1985). Cognitive coping strategies. In Chipman and Segal (eds) *Thinking and Learning Skills*. Hillsdale,N.J.: Lawrence Erlbaum.

Birch, A. (1986). *The Logo Project Book*. Cambridge,Mass.:Terrapin,Inc.

Clements, D. H. (1986). Logo and cognition: A theoretical foundation. *Computers in Human Behavior*,2, pp 95-110.

Clements, D. H. & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology*, 76, 7, pp 1051-1058.

Daiute, C. (1986). Physical and cognitive factors in revising: Insights from studies with computers. *Research in the teaching of English*. Urbana,Ill.:NCTE.

English, R. & Edwards, G. (1984). Programming as a writing activity. *The Computing Teacher*, 11, 6, pp 46-47.

Flower, L. et al (1985) Diagnosis in revision: the experts' option. CDC Technical Report No. 27. ED266464.

Freedman, S. (1985). *The Acquisition of Written Language: Response and Revision*. Norwood,N.J.: Ablex.

Goldenberg, E. P. & Feurzeig, W. (1987). *Exploring Language with Logo*. Cambridge,Mass.:MIT Press.

Graves, D. (1983). *Writing: Teachers and Children at Work*. Portsmouth,N.H.: Heinemann.

Harvey, B.(1986). *Computer Science Logo Style Volume 2 Projects, Styles, and Techniques*. Cambridge, Mass.:MIT Press.

Johnson, R., Johsons, D. & Stanne, M. B. (1986). Comparison of computer-assisted cooperative, competitive, and individualistic learning. *American Educational Research Journal*, 23, 3, pp. 382-392.

Lesser, M. (1985). *Logo for Micros*. London, England:Butterworth & Co.

Papert, S. (1980). *Mindstorms:Children Computers* and Powerful Ideas. *New York:Basic Books*.

Sharples, M. (1985). *Cognition, Computers and* Creative Writing. *New York,N.Y: John Wiley &* Sons

Newkirk, T. (1985). Writing and programming: Two modes of composing. *Computers, Reading and Language Arts*. Winter, pp 40-43.

LOGO, LISTS, AND TREASURE HUNTING:
THE USE OF MULTIPLE CONCRETE MODELS IN THE TEACHING OF COMPUTING

Karen Swan
Robert P. Taylor
John B. Black

Center for Intelligent Tools in Education
Teachers College, Columbia University
New York, New York 10027

Abstract

A mismatch between concrete thinking and the highly abstract nature of programming makes learning to program a computer difficult, especially for children. Analogical reasoning offers one means for bridging this gulf of understanding. In particular, the instructional use of concrete models of computing processes helps beginning students develop the corresponding mental models needed for creative problem solving in complex programming environments. We argue that the instructional presentation of multiple *models will be more beneficial than the use of* single *analogs, because multiple models: 1) develop the differing perspectives necessary for understanding complex ideas, 2) are accessible to a greater range of students, 3) guard against mistakenly incorporating inappropriate features of a model, and 4) develop differing contexts from which general concepts are more easily extrapolated. To illustrate these ideas, we conclude with specific examples of instructional activities designed to model list processing concepts in the Logo computer language.*

The Problem

Most teachers of programming find that once the basic commands of a language are mastered, all but the most able students are stymied by an inability to adequately conceptualize the abstract ideas involved in programming a computer. Often, this gulf of understanding is never bridged.

Kurland, Clement, Mauby and Pea (1984) studied high school students learning Logo in an intensive summer program. They report that most students did not develop enough understanding of Logo to go beyond 'effects generation', that a majority of students had especially poor understandings of powerful abstract structures including variables, recursion, and proceduralization.

Similarly, an intervention devised by Dalby, Touriaire, and Linn (1984), involving junior high school students and BASIC programming, resulted in students learning to solve programming problems whose problem statements exactly matched those used in instruction, but *not* problems that were stated differently or that were more complex. They conclude that students failed to generalize the abstract ideas presented.

Most people's minds do not work well with abstraction. Children are especially concrete thinkers, but even adults typically gain formal understanding of only the particular domains with which they are intimately involved, if they gain it at all. (Ginsburg & Opper, 1979) Programming is a highly abstract process. While computers can indeed be an ideal medium for concretizing abstract ideas (Papert, 1980), this is not to say that programming concepts themselves are at all transparent. The conceptual tools needed for effective problem solving in programming environments typically elude students' understandings.

The literature on domain-specific problem solving emphasizes a relationship between domain expertise and the development of deep structural representations of that particular subject. (Greeno & Simon, 1984). If we would successfully teach programming, we must find ways to help students develop such formal representations of computing.

Developing Mental Models

One path to formal understanding involves the use of mental models, internal analogical representations of complex processes. Mental models help us understand complex systems by facilitating the mental simulation of their operation. Roger Shepard (Shepard & Copper, 1982), for example, has shown that subjects use a mental analog for physical rotation when reasoning about rotated figures. Seymour Papert (1980) speaks eloquently of how his early fascination with differential gears produced a mental model on which his later understanding of mathematics was based.

We can help students develop mental models of complex processes by provided them with concrete analogs for these. Halasz and Moran (1983) report that subjects taught to use a concrete model of the operation of a stack calculator performed better on tasks requiring creative solutions than did subjects who learned to use the calculator without the model. What is most interesting about these results is that the use of the model was specifically helpful in the area of *creative* problem solving, exactly the area where beginning programming students experience the most difficulty.

Indeed, Richard Mayer (1981) conducted a series of experiments to determine the effects of the use of concrete models for programming processes. He provided one group of subjects with diagrams representing input as a ticket window, output as a message pad, memory as a score board, and so on. A control group was given the same information without the diagrams. His results parallel Halasz and Moran's: there was no difference between groups on routine problems, but the students given the models outperformed students in the control group on problems requiring creative solutions.

It seems likely that programming students' common failure to develop adequate conceptualizations of more complex computing processes might similarly be met by providing them with explicit models for these. Our experiences with teaching programming support such a view. We believe, moreover, that the most effective use of analogical modeling will be iterative, that is, that it will involve the presentation of multiple models.

Multiple Models

Modern information processing theories of cognition generally recognize the need to posit multiple internal representations of information in order to adequately account for human performance. (Rumelhart & Norman, 1983) It is not unreasonable to expect that instruction should correspondingly develop multiple representations of complex ideas.

In particular, we maintain that the instructional use of multiple models is desirable for four reasons:
(1) because complex ideas are best represented from a variety of perspectives;
(2) because different models are meaningful to different people;
(3) because the use of multiple models guards against mistakenly incorporating nonrepresentative features of a model; and
(4) because multiple models provide multiple contexts from which particular concepts will most likely be generalized.

To begin with, a variety of models more adequately represents complex concepts than does a single model. Hutchins, Hollan and Norman (1986, p 98) write, "When we deal with the conceptually complex objects of computation, we have to remember that each object typically has a number of different attributes. An object can be viewed from different 'perspectives,' behaving quite differently with each viewpoint. We should recognize that more than one view of an object is often necessary and provide a separate representation for each one."

Secondly, simply because different people bring differing experiences to the learning process, different models are more or less meaningful to them. Papert's gear model, for example, is only useful to someone already familiar with differential gears. Different people, moreover, have different learning styles that are more or less suited to varying representations. With respect to programming, Nancy Cuniff (1986) has found that students with good spatial abilities derive more benefit from the use of a graphical programming environment than do more textually oriented students. Similarly, the more traditional mathematical models of programming disproportionately benefit students with good mathematical skills.

Thirdly, the use of a model always brings with it the danger that students will assimilate more of its features than intended. Computer teachers, for example, frequently compare files saved on disk to common office files. This model fails to capture the fact that computer files are *copied* into memory and exist simultaneously in both places, thus, beginning students often mistakenly assume that files are *moved* from one place to the other. The use of multiple models guards against such mistaken assumptions.

Finally, the use of multiple models to represent abstract concepts makes the desired generalization of those concepts more likely. Gick and Holyoak (1983), for example, found that subjects presented with problem solutions embedded in *two* story contexts transferred the solution to a third analogous problem, whereas subjects who had encountered the solution in but a *single* story context did not.

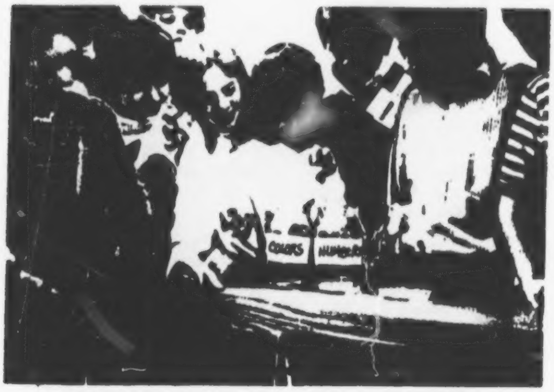
It seems learning is context dependent. Our minds have no way of distinguishing what is, from what is not significant in individual cases. When an idea is encountered in two or more contexts, however, context drops away and concepts are more clearly delineated. Multiple models thus more fully support the development of abstract concepts from concrete analogies. Students in our introductory programming courses at Teachers College, for example, are required to simultaneously learn three languages: FPL, BASIC, and Pascal. (Taylor, 1982) We have found that students so presented with multiple specific contexts are more likely to grasp fundamental computing ideas, and not, as is frequently the case, memorize specific "recipes".

We believe, then, that the use of multiple concrete models of programming concepts can improve the teaching of computing by providing students with an analogical bridge which facilitates the development of corresponding abstract ideas. In a like manner, we believe that providing educators with multiple concrete models of instruction will facilitate their understanding of the concepts involved. We thus conclude this paper with several examples of ways in which list processing concepts in the Logo computer language can be concretely modeled.

List Analogs

Many teachers of Logo believe that list processing ideas are too conceptually difficult for young and/or naive students, and so leave the introduction of these to advanced programming courses. We believe this is unfortunate because list processing is integral to the structure of Logo, hence, an understanding of list processing is integral to an understanding of the language. (Swan, 1986) Moreover, like Bruner (1963), we believe that almost anything can be taught to almost any student given the right representation. We believe the right representation involves the use of multiple concrete models.

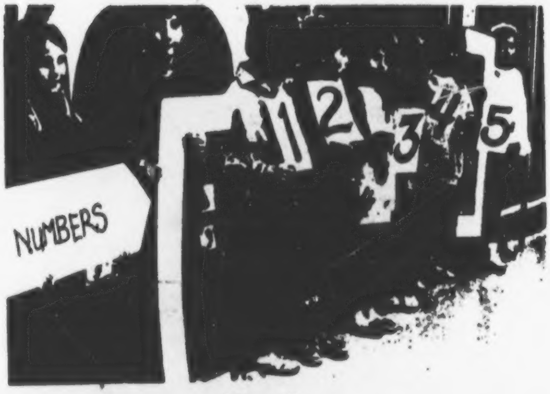
At a most basic level, for example, lists can be represented as *lists*, as the vertical, paper, shopping-list-like lists with which children are most familiar, thus, establishing a link between a familiar concept (shopping lists) and the unfamiliar concept of lists as data structures. We create such physical lists with students. To illustrate what is happening inside the computer when we *MAKE* lists, we make labels for jars representing memory locations (Taylor, 1982), and store the lists in the jars. (Fig. 1) Using paper lists and jars, we can model list creation, list access, list mutation. This particular analog is especially useful for establishing the difference between *variable names* and *variable values*, an abstract and difficult concept for young children, but one they must understand to successfully manipulate lists on the computer.



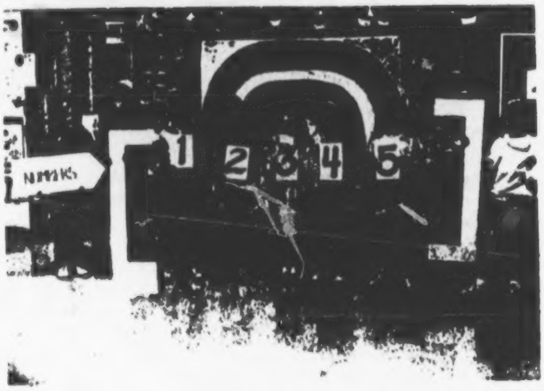
(Fig. 1)

In another activity, students play the various elements of a list. They are given octag signs to hold indicating the name of the list element they represent and placed in a line in the order that such a list would appear if read out on a computer screen. To complete the analogy, students holding octag brackets are placed at either end of the line. (Figs. 2 & 3) Thus arrayed, students can act out the effects that various list manipulations, *FIRST*, *BUTFIRST*, *FPUT*, etc., will have on the list as it appears on the screen. In more complex versions of the activity, students compete to see who can correctly model combinations of commands, and challenge each other by alternatively giving and performing command sequences. We have also developed a solitaire version of the game for individual or pairs of students, that uses note cards and small cardboard brackets.

Just as *playing turtle* helps students develop mental models of graphics commands in Logo by providing them with concrete analogous experience (Papert, 1980), *playing list* helps students develop mental models of list processing primitives by providing them with concrete experiences analogous to list operations. This model is also useful in establishing a link between what is happening in the computer and what is happening on the computer screen.



(Fig. 2)

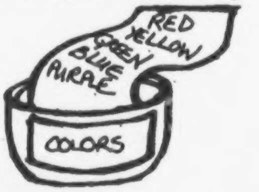


(Fig. 3)

What is happening on the computer screen is, of course, itself a symbolic model of the actual processes taking place inside the computer. We include on-line demonstrations with modeling activities, and extend these latter to student exercises supported by graphic representations. (Figs. 4 & 5) We hope that this interweaving of representations of differing abstractness will lead students from concrete models through iconic and symbolic representations to the highly abstract mental models of computing processes that characterize expert programmers.

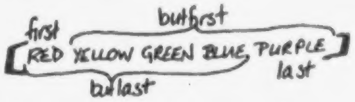
At this highly abstract level, an understanding of what is happening inside the computer revolves around the concept of *linked lists*. Linked lists are the data structures which support the Logo language, and are thus opaque at its user interface. One does not have to understand linked list mechanics, then, to become reasonably adept at

Type: MAKE 'COLORS [RED YELLOW GREEN BLUE PURPLE]



Type: PRINT :COLORS
Draw the screen.

Type PRINT :COLORS
Type PRINT FIRST :COLORS
Type PRINT BUTFIRST :COLORS
Type PRINT LAST :COLORS
Type PRINT BUTLAST :COLORS



Draw the screen.

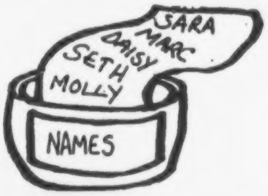
(Fig. 4)

programming in Logo. Such understanding, however, becomes increasingly useful with increasing expertise. Moreover, the ideas involved in linked list operation can be quite elegantly modeled using the *treasure hunting* game.

The game of treasure hunting is a favorite with children, thus both familiar and fun. It begins with a clue that sends players to a location where a second clue is hidden. The second clue sends them to a second location where a third clue is hidden which sends them to a third location where a fourth clue is hidden, and so on. The *treasure* in treasure hunting is found at the end of this trail of clues. (Fig 6)

The clues in treasure hunting are analogous to the pointers which connect linked lists inside the computer. A clue can be *inserted* in the treasure hunting trail simply by exchanging the directions to its hiding place for those found at the spot from which it will follow, and placing these at the new location. A clue is *deleted* by exchanging its directions for those which pointed to it. The game can be made to even more closely resemble the operation of linked lists by having each location contain both the *address* of the next clue and some sort of information, a part of a famous quotation

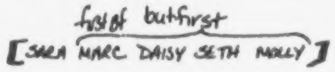
Type: MAKE 'NAMES [SARA MARC DAISY SETH MOLLY]



Type: PR :NAMES
Draw the screen.

Type PR :NAMES
Type PR FIRST BF :NAMES
What happens? ____________________

____



Draw the screen.

(Fig. 5)

(Fig. 6)

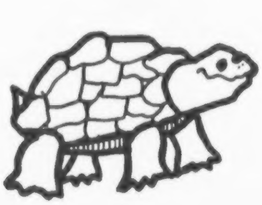
for example, which is assembled by traversing the clue trail. Continuing the analogy, garbage collection can be modeled as the removal of deleted clues, and active clues can be deliberately removed from the trail to illustrate the disastrous consequences of lost pointers.

The instruction predicated on these three concrete analogs for list processing illustrates the advantages of the use of multiple concrete models in the teaching of computing. Firstly, the three represent differing, and at least in the case of the first two, necessary perspectives on list processing. Lists are created as variables (paper lists and jars) and manipulated internally (playing lists). The mechanisms supporting such processes involve pointer control (treasure hunting). These three quite different perspectives all add something to a mental model of list manipulation, and taken together obviously present [illegible] more complete picture of list processing than could a single model alone.

Secondly, the instruction built around these models includes not only the more obvious differing contexts, but differing levels of abstraction presented via differing media as well. It stands to reason that such greater diversity provides a greater variety of access to the fundamental list processing concepts for students who will bring a variety of prior experiences and learning styles to the learning experience.

Thirdly, it is easy to see how any one of these models taken by itself could be misleading. The treasure hunting model, for example, could lead one to believe that the information stored in lists is found at the end of its trail of pointers, when, on the contrary, all that is really there is the *NIL* marker. Similarly, playing lists could create the impression that list elements are actually moved around in memory, when, in fact, it is the pointers which are changed. The use of multiple models, however, makes such mistaken assumptions unlikely, as they are contradicted by at least one of the other models.

In a like manner, the use of multiple models helps keep the concepts embedded in these multiple contexts from remaining context bound. One is clearly more likely to extrapolate a general understanding of list processing from many contexts than from a single one. In fact, in the classroom, the instructional use of these three models seems highly effective. We have successfully used them to introduce basic list processing to students aged six to sixty.



References

Bruner, J. (1963). The Process of Education. New York: Vintage Books.

Cunniff, N., Taylor, R. P., & Black, J. B. (1986). Does programming language affect the type of conceptual bugs in beginners' programs? CHI '86 Proceedings. Boston, MA.

Dalby, J., Tourniare, F. & Linn, M. (1984). Making programming instruction cognitively demanding. (Tech Report 29). New York: Center for Children and Technology, Bank Street College of Education.

Ginsburg, H. & Opper, S. (1979). Piaget's Theory of Intellectual Development. Englewood Cliffs, NJ: Prentice Hall.

Gick, M. L. & Holyoak, K. J. (1983). Schema induction and analogical transfer. Cognitive Psychology, 15, 1-38.

Greeno, J. G. & Simon, H. A. (1984). Problem solving and reasoning. (Technical Report No. UPITT/LRDC/ONR/APS-14). Pittsburg: University of Pittsburg, Learning Research and Development Center.

Halasz, F. G. & Moran, T. P. (1983). Mental models and problem solving in using a calculator. CHI '83 Proceedings. Boston, MA.

Hutchins, E. L., Hollan, J. D. & Norman, D. A. (1986). Direct manipulation interfaces. In D. A. Norman & S. W. Draper (Eds) User Centered System Design. Hillsdale, NJ: Erlbaum.

Kurland, D. M., Clement, K., Mawby, R. & Pea, R. D. (1984). Mapping the cognitive demands of learning to program. (Tech Report 29). New York: Center for Children and Technology, Bank Street College of Education.

Mayer, R. E. (1981). The psychology of how novices learn computer programming. ACM Computing Surveys, Vol. 13, No. 1, 121-141.

Papert, S. (1980). Mindstorms. New York: Basic Books.

Rumelhart, D. E. & Norman, D. A. (1984). Representations in memory. (Technical Report No. ONR 8302). La Jolla, CA: Institute for Cognitive Science, University of California, San Diego.

Swan, K. (1986) "Primarily Lists." Logo '86 Pre-Proceedings. Cambridge, MA: MIT.

Taylor, R. P. (1982). Programming Primer. Reading, MA: Addison-Wesley.

EMPOWERING THE TEACHER... *CREATING A LOGO CULTURE IN LESS THAN IDEAL SITUATIONS*

Gary S. Stager
Director of Training: Network for Action in Microcomputer Education
Consultant/Teacher-Trainer: Logo Computer Systems, Inc.
Adjunct Professor: Kean College of New Jersey

Abstract:

Many of us agree that the Logo programming language and accompanying educational philosophy has the potential for influencing profound positive educational change. Is this change possible only in "Utopian" research settings or can Logo learning be achieved in an average public school with a shortage of microcomputers and technologically inexperienced teachers?

I believe that we ***can*** Empower Teachers to create Logo cultures that nurture Logo learning in the less than ideal situations many of us find ourselves in today. In fact, it is our responsibility as educators to prepare schools for the future developments in educational computing.

I propose that in order for us to empower the learner, we must first empower the teacher. There are several ways in which teachers can be assisted in developing a Logo culture, these include; adequate teacher-training, simple and interesting Logo project ideas, the project approach to learning, Logo curriculum connections, involving children in the evolution of a classroom Logo culture, training students to be Logo resources, giving students responsibility for sharing knowledge with other students and teachers, peer tutoring, classroom integration strategies, and ways in which teachers may share information.

In this presentation I will share Logo project ideas and will show examples of student's work with LogoWriter™. I will also reflect on my four years of experience training teachers and children how to use Logo to enhance learning and revitalize the curriculum.

Introduction:

One of the most profound goals of the Logo movement is to empower the learner. This concept, discussed by Seymour Papert, Jean Piaget, and others, is used to stress the importance of an individual (in our case a student) to take responsibility and control over his/her own learning. There are many positive implications for the empowered learner; self confidence, realization of one's abilities, concretization of the abstract, intrinsic motivation and aesthetic intent for learning, internalizing knowledge, accomodation of different learning styles, and appropriation of knowledge. Logo provides a context for empowering more learners than most other educational tools or philosophies.

For better or worse, the key figure in American schools is the teacher. If Logo is to succeed in our schools, we need to accomodate the needs, fears, abilities, and restrictions of the classroom teacher. It is for this reason that in this presentation I propose to illustrate how important it is for us to empower the Logo teacher.

My presentation will be divided into two main topics. The first is that educators need to be provided with "powerful ideas" that can be implemented with little Logo programming, as well as, models, tools, and materials to expand their skill as a Logo teacher/learner. The second idea is that students can and should play at least as large a role as the teacher in contributing to the formation of a Logo culture. I will provide practical Logo ideas, as well as share my thoughts and experiences of how Logo can be a positive educational force, despite the typically less than ideal realities of most traditional classrooms.

Tools for the Teacher:

I intend to provide examples of how small ideas can be expanded into large scale Logo projects, in a variety of curriculum areas. I will stress how Logo can be used to create microworlds that have connections to the traditional curriculum and how Logo activity may cause us to reflect on how we think about and teach in these curriculum areas. Logo seems to be a powerful diagnostic tool for gaining a more insightful understanding.

In my opinion, the goal of educational computing is to make the microcomputer a transparent educational tool. If you ask a poet deep in thought what she is doing, the answer will not likely be "pencil". By the same token, we should be able to ask a student sitting at a micro, "What are you doing?" Today, most kids will respond, "Computer", "Logo", or "Word Processing." When children are comfortable with the technology and have greater access to microcomputers they should respondby saying I'm "writing a story", "drawing a picture", or "solving a math problem"; tasks firmly rooted in their personal experiences.

One way of knowing if the microcomputer is being thought of as more than a discrete subject of study is when we begin seeing examples of the technology being embraced by individual students. Students are embracing the technology when they appropriate the tool as a natural vehicle for doing things with the computer we might not expect. For instance, high school students writing illicit love letters to their friends, working on projects in other curriculum areas, or a student who used LogoWriter to create an animated story of the seven days of creation. These and many other examples provide a vision of what educational computing can be when the logistical problems of limited hardware and access are resolved. Although it will take a few years to reach a point when school conditions support widespread appropriation of educational techology, educators must begin to create classroom environments which nurture the spirit of Logo and accessible technological tools.

Teachers too often are dependent on inferior and irresponsible Logo books and materials or have time constraints imposed that don't allow them to pursue intensive Logo training. Many are unaware of Logo's potential as a tool for interdisciplinary learning. This lack of accessible resources and knowledge may cause educators to be unaware of practical "do-able" ideas for Logo activities and therefore limit the long-term potential of Logo due to their inability to make Logo relevant in their classrooms.

There is hope for these teachers. Simple Logo project ideas can be used as starting points for any teacher's use of Logo in their classroom. These powerful ideas serve as intellectual models which can be expanded and adapted to meet the needs of students. The ability to create "objects to think with" provides the teacher with a magical palette of powerful ideas at the tip of his/her fingers. This new sense of the "teacher as learner" empowers the teacher, as well as, their students. During my presentation, I will share simple Logo ideas which have powerful implications for exploring mathematics, language arts, science, social studies, music, and art. (LogoWriter™ will be used to illustrate many of these ideas).

One of the most powerful applications of creative Logo use is the creation of student projects. The "Project Approach" to Logo allows children to become immersed in a project that is not only educational, but of signifigant interest to the student. When a student takes on a project he/she confronts a number of learning issues and problems that need to be solved. The student learns to plan a project, set goals, and enhance their individual social skills through teamwork and the division of responsibility. The Logo project brings the subject matter to life and nearly every Logo project is interdisciplinary in scope. As a result of the "Project Approach" a student appropriates a new body of knowledge that could not have possibly been provided by the traditional curriculum. It is nearly impossible to anticipate the vast spectrum of knowledge acquired by learners through the project approach.

I will share simple Logo ideas which have powerful implications for exploring mathematics, language arts, science, social studies, music, and art. The purpose of this part of the presentation is to demonstrate the educational potential of Logo more than understanding the nuts and bolts of the individual examples.*(LogoWriter™ will be used to illustrate many of these ideas).* Some of the following project ideas may be demonstrated and discussed:

Mathematics:
A Fractions "Calculator" Microworld for performing arithmetic operations on fractions, New Math Predicates and Operations, Solving Word Problems with simple procedural machines, Exploring geometric concepts, Counting and Measuring with the turtle, Graphing, Baby Turtle procedures, Exploring simple Probability and Statistics.

Social Studies:
Drawing maps(figure 3), Understanding scale, Designing Communities(figure 9), Creating Timelines, Creating Pie Charts and Bar Graphs(figures 5 & 6), Family Trees, Designing Datbases, Using LogoWriter to Explore Elections; as a voting machine, for creating campaign "commercials", and for graphically displaying results, Creating illustrated and animated reports.

Language Arts:
Building your own word processor, Creating a large letter word processor, Building Spelling and Grammar Checkers, Writing illustrated and animated stories(figures1,4, & 9), Creating Adventure Stories, Designing Greeting Cards(figure 3), Cartoons, and Turtle Plays, Dynamic and Interactive Text.

Science:
Exploring simple Physics through Animation, Plotting the results of External Probes (figure 7), Simulating Animal and Robotic Behavior (figure 1), Gears and Meters(figure 2), Creating Models of Planets and Constellations, Exploring Vectors.

Art:
Quilts, Drawings(figure 10), Animations(figure 4), T- Shirt Designs(figure 10).

Music:
Understanding Pitch and Duration, Manipulating Music, Connections between Music and other Subject Areas, Creating and Composing with Tune Blocks.

The Role of the Teacher and Student in the Creation of a Logo Culture:

I don't believe that Logo is a discrete subject to be taught through drill in a segregated computer lab. In order to have an impact on how students learn and think, Logo must be a readily accessible tool *in* the classroom. This makes Logo relevant to interdisciplinary learning. Given the unfortunate realities of most schools; lack of computers, inadequately trained teachers, growing pressures and responsibilities being placed on teachers, and a finite number of hours in the school day, we must devise creative strategies for creating a positive Logo culture in a traditional classroom. I agree with Seymour Papert who recently said that, " although computers are not as accessible to students today as they should be, in five or ten years there will be a high density of computers in most schools and we must begin to plan now for how we are going to use that technology." We must make the best of our limited resources and begin to envision the future, today.

At Logo '86, Seymour Papert described Logo as a "Trojan horse for educational change." I submit that there needs to be a **"Trojan horse" for the "Trojan horse"** - at this point in time, children need to provide much of the impetus for the evolution of a Logo culture. Teachers need not be the only sources of knowledge in the classroom. Students can play an important role in the transmission of knowledge and creation of a rich learning environment. On the surface, students are merely assisting the teacher in influencing educational change. The real benefits of this involvement are much more profound. There are a multitude of ways in which the student may be empowered by being a participant in the creation of this Logo culture.

I would like to share a model I have been developing in a New Jersey elementary school. I have assembled a group of students who serve as Logo leaders in their school. Regular classroom teachers select 2-4 students from each fourth and fifth grade class and every couple of months I bring them together for a half-day Logo workshop. These workshops are handled in much the same way as teacher-training workshops.

Some students have had a fair amount of Logo experience and some have barely used computers due to a lack of a coherent district plan. To help reduce this disparity, the workshops began with very elementary Logo concepts (new for some, review for others) and I attempted to pair novices with more experienced programmers.

The content of these workshops is geared towards getting these student leaders over the next Logo hurdle, (defining procedures, variables, interactive programming, animation, recursion, combining text and graphics, etc...) as well as, sparking some creative ideas that will sustain Logo activity until the next workshop. If students can tell me in English what they wish to accomplish, I will show them the Logo code. I always strive to seize the teachable moment.

The students involved in this program understand that they have a responsibility to help the other students (and even teachers) learn Logo and are becoming quite competent sources of information. Students and their teachers have taken it upon themselves to regularly teach Logo to other classes of children and this has naturally extended the Logo culture into the younger grades.

This program has met with great encouragement from students, teachers, parents, and administrators. The educational benefits for the students are many; students experiening the role of learner/teacher, responsibility for one's learning, transmission of knowledge, reflection on one's own thinking (metacognition), filling in fractured knowledge, greater self-confidence, and the expansion of an educational environment over time. There are also practical benefits for the school, namely Logo learning might have been abandoned or less profound had all of the responsibility for implementation been placed on the shoulders of overworked teachers or ambivalent administrators.

In no way do these ideas preclude an interested teacher from taking an active role in using Logo with his/her students. All teachers are encouraged to participate in this program and are provided with any needed training to support their activities. The minimum requirement for a teacher participating in this program is that he/she is at least flexible enough to allow children to be independently working with Logo while he/she is doing something else. I've found that by making students the key figures in this culture, teachers see the value of Logo and take a more active interest in nurturing their students' Logo activity.

Formal Logo teacher-training must achieve a delicate balance between hands-on experience and discussion of the Logo philosophy. My primary goal in teacher-training is to create situations in which teachers can achieve early success at the computer. If the teacher is successful in his/her hands-on time, they are more likely to be enthusiastic and self-confident when they introduce Logo into their classroom. I supplement the hands-on instruction with discussions and readings on Logo philosophy, implementation, and research.

I am currently very enthusiastic about my work in developing a Logo culture in a first and fifth grade classroom and the effects of the LogoWriter Home License on the acceleration of the evolution of a school Logo culture. The unprecedented ability to have students take home copies of Logo promises to extend the Logo culture into the outside world and provides students with much more hands-on time with Logo.

I will also discuss possible resources for teachers to support one another in their Logo activities. These resources include journals, workshops, users groups, and on-line conferencing.

Conclusion:

During the past three school years, I have experienced encouraging success in creating a Logo culture in elementary schools that would not generally be thought to support the the incorporation of the Logo philosophy. I will discuss models for Logo teacher-training and ways in which it is possible to provide children with a meaningful Logo experience (in the spirit of Logo exploration) with a limited number of computers. I believe that Logo is too important an educational tool/philsophy to abandon just because of less than ideal school conditions. The act of true courage in this educational revolution is to make Logo work despite the odds, not to retreat. We must empower our teachers if we are to empower our students.

Handouts containing Logo project ideas and implementation strategies will be available at my presentation.

References

Abelson, Harold, Andrea diSessa. Turtle Geometry: the Computer as a Medium for Exploring Mathematics, Cambridge, Mass.: MIT Press., 1981.

Clements, Douglas. "Logo Programming: Can it Change How Children Think?" Electronic Learning (January 1985).

Gelman, Michael. "Simulating Robotic Behavior Using LogoWriter." Preceedings - The East Coast Logo Conference 1987, Westport, Connecticut: Meckler Publishing Corporation., 1985

Goldenberg, E. Paul. "Iconic Programming: A Strategy and Metalanguage that Helps One Learn the "Hard Parts" of Logo." from The Logo '86 Conference Proceeding, Cambridge, Ma.: M.I.T., 1986.

Harvey, Brian. Computer Science Logo Style: *Intermediate Programming*. Cambridge, Mass.: MIT Press., 1985.

Harvey, Brian. Computer Science Logo Style: *Programming Projects*. Cambridge, Mass.: MIT Press., 1986.

Huberer, Leonard N. "Computer Learning through Piaget's Eyes." Classroom Computer Learning (October 1985).

Leron, Uri. "Logo Today: Vision and Reality." The Computing Teacher (February 1985).

Papert, Seymour. "Computer as Mudpie." from Intelligent Schoolhouse: Readings on Computers and Learning, edited by Dale Peterson. Reston, Va.: Reston Publishing Co. Inc., 1984.

Papert, Seymour. "Computer Criticism vs. Technocentric Thinking." from the Logo '85 Theoretical Papers. M.I.T. Learning and Epistemology Group. 1985

Papert, Seymour. "Different Visions of Logo." from Computers in the Schools. Haworth Press Inc., 1985

Papert, Seymour. "Interview with Seymour Papert." Omni Magazine, October 1985.

Papert, Seymour. "Logo as a Trojan Horse: Rethinking Logo Experience in the Context of a Real School Setting." from The Logo '86 Conference Proceeding, Cambridge, Ma.: M.I.T., 1986.

Papert, Seymour. "New Theories for New Learnings." School Psychology Review, Volume 18. Number 4. 1984

Papert, Seymour. Mindstorms: Children, Computers and Powerful Ideas. New York: Basic Books, Inc., 1980.

Rosen, Marian. "But Wait- There's More!" from The Logo '86 Conference Proceeding, Cambridge, Ma.: M.I.T., 1986.

Solomon, Cynthia, Brian Harvey, Margaret Minsky. Logoworks. New York: Byte/McGraw-Hill., 1986.

Stager, Gary S. "If There Had Been Logo, I Might Have Learned Math." from The Logo '86 Conference Proceeding, Cambridge, Ma.: M.I.T., 1986.

Stager, Gary S. "The Literate Turtle - Novel Language Arts Applications in LogoWriter" (Unpublished), Wayne, NJ: 1987

Stager, Gary S. "Logo and Music - A Powerful Tool for Learning." from The Logo '86 Conference Proceeding, Cambridge, Ma.: M.I.T., 1986.

Taylor, Robert P. (editor). The Computer in the School: Tutor, Tool, Tutee. New York: Teacher's College Press., 1980. (Papert articles)

Thornburg, David D. Beyond Turtle Graphics. Menlo Park, Ca. :Addison-Wesley., 1986.

Thornburg, David D. "Logo: A Computer Language At The Crossroads" A + Magazine. (March 1986).

Thornburg, David D. "The Left Hand of AI: Bringing Design Students to Logo." Transcript from the 1985 World Logo Conference. Compuserve's Logoforum.

Turkle, Sherry. The Second Self: Computers and the Human Spirit. New York: Simon and Schuster (Touchstone)., 1984.

Watt, Daniel. Teacher-Made Microworlds: or Training Teachers to Use Logo vs. Training them to Teach It. Antrim, New Hampshire: Educational Alternatives.

Watt, Molly. "Twenty Powerful Ideas" National Logo Exchange (September 1985).

Watt, Daniel, Molly Watt. Teaching with Logo: Building Blocks for Learning. Menlo Park, Ca.: Addison-Wesley., 1986.

Watt, Daniel, Molly Watt. "Logo Quilt: A Collaborative Learning Project" National Logo Exchange (May 1986).

"Computers May Play Key Role in Traditional Music Studies." The New York Times. New Jersey Section, page 20. July 8, 1984.

GARY S. STAGER

Computer Education Consultant
Teacher-Trainer

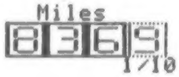
12 Locust Place
Wayne, N.J. 07470
201-831-0133

SUE

Once upon a 🕒 there was a 👧 named Sue. Sue lived in a red 🏠 by a great big Oak 🌳. She has a 🐈 named Fluffy and a 🐇 named Bugs. Sue ❤s her pets, her house, and her parents very much.

Figure 1

This is an example of a digital odometer, like the ones found in automobiles.


Miles
8369
1/10

This program uses four turtles and animates the oddometer by changing each turtle's shape from shapes 21-30.

Figure 2

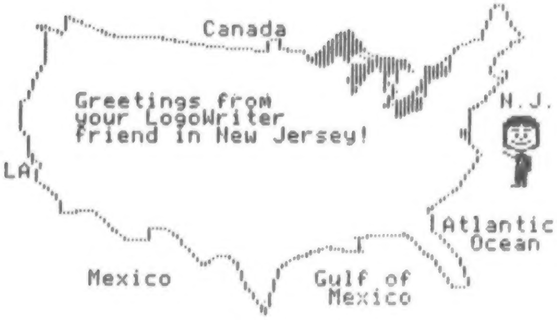
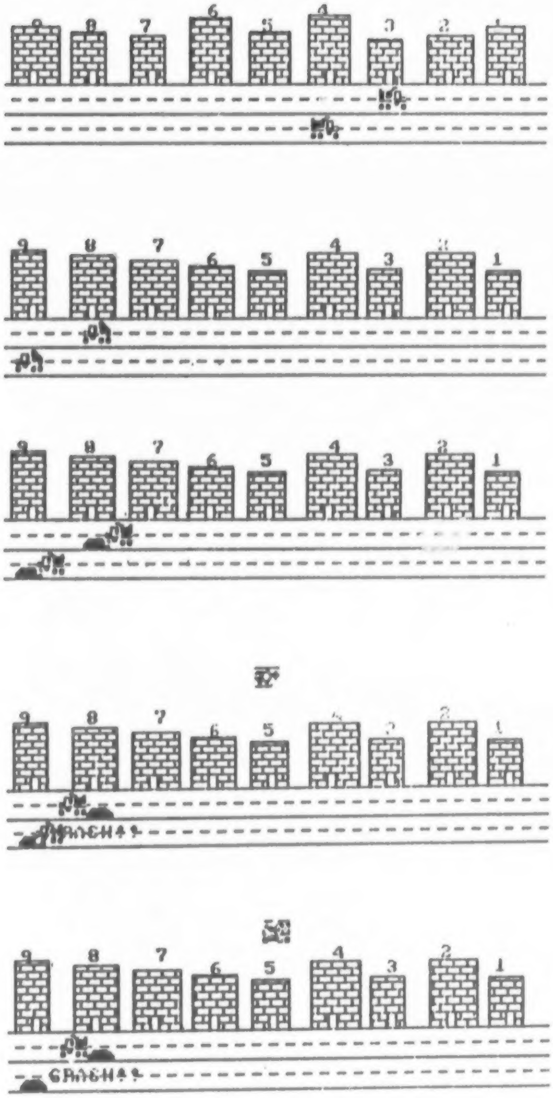
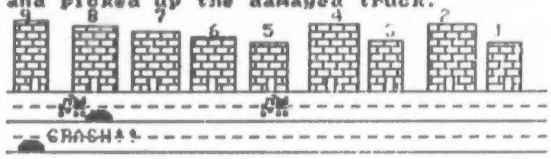

Canada
Greetings from your LogoWriter friend in New Jersey!
N.J.
LA
Atlantic Ocean
Mexico
Gulf of Mexico

Figure 3


9
8
7
6
5
4
3
2
1
CRASH!!

...One of the trucks crashed into a pile of dirt it had dropped in the road. An emergency helicopter responded and picked up the damaged truck.


9
8
7
6
5
4
3
2
1
CRASH!!

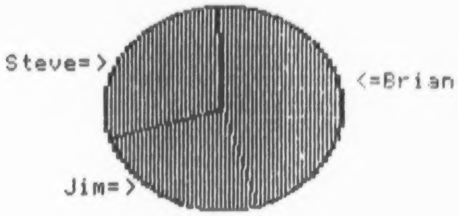
The truck was repaired and brought back to the highway.

By: Michael Gelman-age 10

Figure 4

Class Election Returns - Class 5S

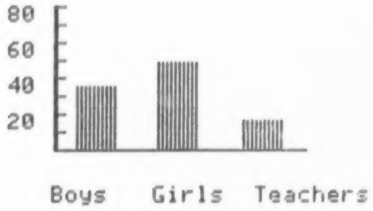
Brian - 11 Votes = 46% of the vote.



Steve - 7 Votes = 29% of the vote.

Jim - 6 Votes = 25% of the vote.

Figure 5



Number of boys, girls, and chaperones at the 8th Grade Halloween Dance.

Figure 6

TEMPERATURE GRAPH

RED IS CELSIUS, BLUE IS FAHRENHEIT

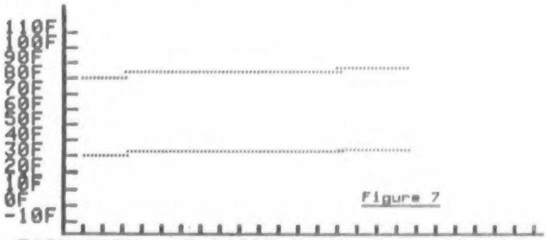
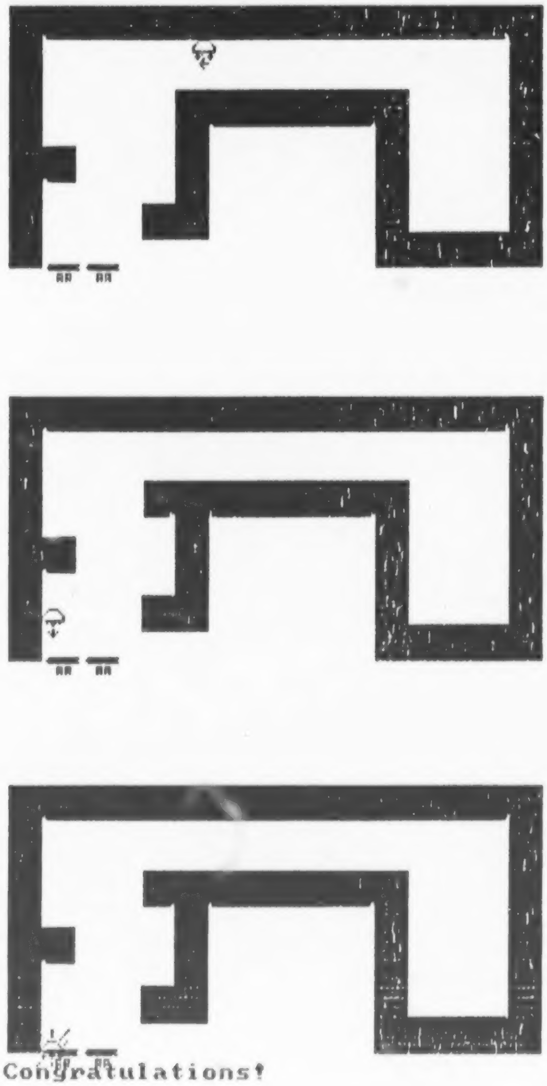


Figure 7

EACH MARK = ELAPSED TIME OF 1 SECONDS

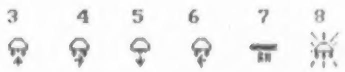


Congratulations!

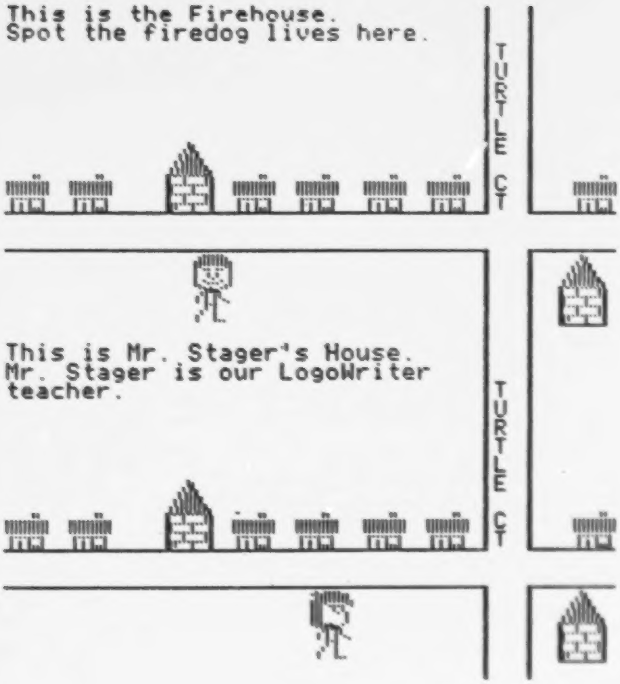
Peppy found its food in 265 moves.

Figure 8

Turtle shapes used in the LogoWriter Robotic behavior simulation.



Shapes 3,4,5, & 6 represent the different headings of the robot while in the maze, Shape 7 is the robot's food, one of two batteries, and shape 8 is the shape of the happy robot when it finds its batteries and eats.

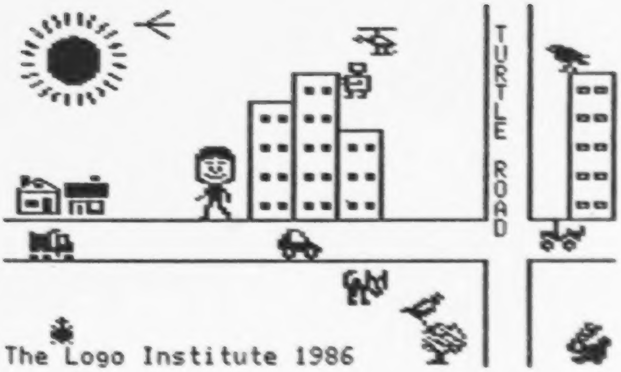


This is the Firehouse.
Spot the firedog lives here.

This is Mr. Stager's House.
Mr. Stager is our LogoWriter teacher.

TURTLE CT
TURTLE CT

Figure 9



The Logo Institute 1986

TURTLE ROAD

Figure 10

COMPUTER ASSISTED HEALTH PROMOTION: CURRENT DEVELOPMENTS

Ernest A. Randolfi, Ph.D. and David Bacharach, Ph.D.

Department of Health and Physical Education
Syracuse University, 820 Comstock Avenue
Syracuse, New York 13244

Abstract

This paper is an examination of representative software which is available for health and physical educators and includes some implications of computer use in health promotion. Current software applications for health promotion can be categorized into three groups: 1. computer based instruction, 2. health and fitness assessments, and 3. computerized health management software. Computer based instructional programs are available for a variety of health topics including but not limited to stress management, drug education, human sexuality, nutrition, exercise physiology, and first aid and safety. Exemplary programs emphasize a variety of educational and behavioral strategies (e.g., decision making, skill development, values clarification and health simulations). Due to the sensitive and sometimes controversial nature of health topics the educator must carefully assess both the content validity and the appropriateness of the educational message.

By collecting and analyzing individual attributes of health behavior, computerized health and fitness assessments expedite and enhance health promotion efforts. Health risk appraisals, diet analyses, fitness assessments, and stress management inventories are commonly used to inform, motivate and personalize health and fitness instruction. A reliance upon normative data in the construction of these applications suggests a need for qualified interpretation of computer output. For instance, with health risk appraisals individual users may not be knowledgeable of how risk factors are derived and may not understand how to effectively reduce their health risks.

The final software category includes programs used to track and manage health behavior progress. The variety of applications that could possibly be listed within this area are indicative of the scope and diversity of computer use in health promotion today. Programs exist which will keep family medical records, log exercise workouts, chart health behavior change strategies, and dictate daily weight loss diets. Like the other applications these programs are only as good as the empirical base from which they are derived. The danger lies in people receiving inaccurate and misleading information on which to base their health decisions. Health and physical education professionals should be knowledgeable of the analyses performed by software applications and should assess the quality of health information being presented by the program. The results of these efforts should ensure better health promotion efforts and a resulting healthier population.

Introduction

This paper represents a survey of computer applications which have been developed to promote health and includes some implications of their educational use. An expanding research base exists to support the use of computers as an educational media in health and physical education. Many educational computer applications utilize quick information retrieval, interactive branching routines, accurate numerical computation, are self-paced, and include interfacing possibilities with other media. Compared to traditional formats, computerized approaches to health education have been shown to be equal or better than other methods on a variety of indicators (Deardorff, 1986; Cicciarella, 1983). Individuals, including those not familiar with computer operation, seem to enjoy receiving their health information through this machine (Bauer & Surman, 1985; Ellis & Raines, 1981).

The response has been so positive, professional health related journals have devoted entire issues to the technological innovation of computers in health promotion (Health Education, 14(6), 1983; Journal of Nutrition Education, 16(2), 1984; Journal of Physical Education, Recreation and Dance, 54(9), 1984; The ACHPER National Journal, 108, 1985; Family and Community Health, 9(2), 1986). Similarly, the importance of this trend is exemplified by increasing numbers of preservice health and physical education programs which are aware of the need to prepare computer using professionals who will know how to use the computer to increase personal effectiveness and promote health (Breckon, 1985; Bailey & Pigg, 1983). A recent survey of 120 graduate health education programs in North America indicated that as many as 25.8 percent were offering educational microcomputer courses for their students (Randolfi, 1986).

The diffusion of computer assisted health promotion is also evident in the extensive collection of commercially available software products. Programs have been developed for all ages and most health related topics (National Health Information Clearinghouse, 1984). Subject matter from first aid to contraception is available in computerized instructional formats. The majority of these programs have been marketed to school age populations and professional health/fitness centers, however, a growing selection of software has become available for the adult home user. As the general public continues to become increasingly involved in the pursuit of health and fitness, more and more people seem to be searching for innovative and efficient means to achieve their health and fitness goals. The available software applications can be grouped into three broad categories that include: 1. computer assisted instruction, 2. health and fitness assessments, and 3. computerized health management software.

Although the quantity of software has grown, health professionals have been critical of the educational quality of this software. McDermott and Belcastro (1983) have suggested that much of the available software has either been written by programmers that don't know much about health education or by health educators that don't know much about programming. While this may be true of the early programming efforts which were simplistic in design, increasingly sophisticated applications are now available for both the health professional and the general public.

Computer Assisted Instruction

Unlike other educational disciplines, health related computer assisted instruction is not limited to the transference of knowledge and facts. The fields of health and physical education are uniquely involved in motivating people to change and/or maintain health enhancing behaviors. The assumption that knowledge alone will be sufficient in encouraging behavior change has long been rejected by health promotion professionals. Health related behaviors are often a product of one's socio-cultural environment and have been learned over the course of one's lifetime. Therefore, it would not be reasonable to expect any twenty minute session of computer assisted instruction (CAI) to effectively change an individual's lifestyle behaviors. Computer assisted instruction can be used to introduce new concepts, supplement existing instruction and reinforce what students already know. What the computer brings to health promotion efforts is an entertaining and innovative form of instructional media which students enjoy using.

Educational strategies for computer assisted health promotion focus on increasing student involvement in the subject matter. Many programs are interactive and rely upon user input to individualize instruction. Other programs utilize the concept of computer games to increase student involvement. Woodward, Carnine, and Davis (1986) found computer games and simulations to be highly effective in the presentation of new health information. Computer games are available to supplement nutrition education, relaxation training, safety instruction, general health, and other content areas.

Along with teaching new information, computer assisted instruction in health and physical education can be used to help teach personal decision making regarding health behaviors. By presenting students with real life vignettes of health behavior decisions, the student is given an opportunity to clarify their values and analyze their decision making ability. Realistic simulations allow the user to view the results of their decisions without having to personally experience the consequences. Existing programs using this strategy include examples of seatbelt safety (Randolfi, 1985) drug, alcohol, tobacco use, and human sexuality.

In addition to promoting decision making strategies, programs have also been developed to help students learn in exciting new ways. The interface of biofeedback instrumentation with the microcomputer offers a useful demonstration of how the microcomputer can teach health enhancing skills. Biofeedback instrumentation is often used in stress management education to measure physiological responses too subtle for normal awareness. By providing immediate visual and auditory feedback of a variety of physiological processes, students can learn to consciously control levels of emotional arousal and subsequently improve their ability to handle stress. In physical education these devices can be used to illustrate exercise physiology principles and to track improvements in fitness levels. Programs are available to interface with instruments that measure heart rate, muscular tension, body temperature, galvanic skin response (Randolfi, 1987), muscular endurance, strength, and body fat.

A number of other software applications are also available for health promotion. As the quantity of health information doubles at a rate of every five years, educators and students will come to rely on the microcomputer as an information retrieval system. Fast access to an extensive library of visual images and auditory recordings will provide a new and stimulating dimension to learning environments. As an information processor, few educational media can match the qualities of an interactive data base of current health information. An excellent example of such a system was developed at the University of Wisconsin (Bosworth, Gustafson, Hawkins, Chewing, & Day, 1983). This system offers nonjudgemental, confidential information, and community referrals on the topics of alcohol and other drugs, human sexuality, smoking, stress management, and body management. In theory this program functions by having a computer placed at a central location in the high school. Students have access to 35 floppy disks which include more than 25 hours of activities and health information. While the program is not intended to replace teachers or counselors, students who are reluctant to ask teachers or their peers questions about personal health have welcomed the opportunity to use such a program. The program has been extensively field tested and has been shown to be effective in altering student attitudes toward risk taking behaviors.

Interactive videodisc and videotape systems exemplify still another way that technological advancements can promote learning in health and physical education (Breckon & Pennington, 1986; Levenson & Morrow, 1986). Interactive videodisc has been successfully used to teach cardiopulmonary resuscitation (CPR) via a self contained system that includes the interface of resuscitation manikins, videodisc, and the microcomputer (Hon, 1982). This system will teach, monitor, and evaluate CPR instruction by demonstrating the correct procedure, allowing the student to attempt the skill, and then telling the student whether or not they performed the skill correctly. Videodisc systems are now being developed for stress management, first aid, nutrition, smoking cessation, and hypertension instruction. Breckon and Pennington (1986) have evaluated the potential of interactive videodisc for health promotion and concluded that, while the cost of such a system is still prohibitive and continued evaluation is necessary, computer assisted video instruction may be the ultimate delivery system for health promotion efforts.

Computerized Health Assessments

The "number crunching" ability of microcomputers is often taken advantage of in many health and physical education software packages. This is especially true in the personalized health assessment applications. These programs offer many powerful innovations to the health professional's arsenal of educational material and include health risk appraisals, nutritional analysis, stress management inventories, and physical fitness assessments.

Health risk appraisals (HRA) are one common example of how computers are being used to promote health on a regular basis. These programs compare health practices of individuals to normative data on morbidity and mortality and provide an assessment of one's potential risk for illness, injury or premature death. Generally, an individual is asked to complete a questionnaire on their medical history and personal health behaviors, these data are then entered into the computer which analyzes the responses and prints out specific health/lifestyle recommendations. Programs have most often been developed for adult populations, but a few HRA have been designed for use with high school and college students. An entire branch of epidemiological science and industry has developed around the use of these instruments as they have become a popular component of health fairs and wellness centers.

Nutritional analysis programs are another popular health assessment tool. In using this application, one enters a list of foods eaten during a 24 hour period into the computer. The quantity and type of food eaten is compared to a data base of food nutrient values and the recommended dietary allowances (RDA) for that individual. Some programs will indicate where deficiency or excess consumption exist, and in some cases will make recommendations for dietary improvement. As a result, dietary instruction/intervention becomes personalized and relevant to the individual.

A third health promotion tool is the computerized physical fitness assessment which is used extensively to assist in the development of individualized exercise prescriptions. Data from exercise and physiological testing (e.g., body fat measurements, aerobic capacity, strength measurements) are entered into the computer and individuals are given a representative analysis of their fitness level. Such programs are popular among school physical educators who routinely evaluate large numbers of students and are able to do so more efficiently with the computer's assistance (Randolfi, 1986). Physical fitness assessments are also used in a variety of other settings (e.g., health clubs and hospitals) as motivational and prescriptive tools for adults.

Another less common assessment tool is the computerized stress management inventory. The best of these use valid and reliable instruments of a psycho-social nature to conduct personality profiles which in turn may indicate possible areas for education/ intervention. Users answer questions on the type of stressors they encounter, how they respond to these stressors, what prevention and treatment strategies they currently employ, and how do they handle personal time management. Used in combination with comprehensive stress management instruction these inventories can be very useful and educational tools.

It should be obvious that if educators intend to use the various health assessment programs described above, they should be prepared to interpret output for their students or provide learning opportunities which address these application issues. These applications are not intended as stand alone products. They are useful, however, when used by trained personnel as part of a larger health promotion plan.

Computer Based Health Management

The variety of applications that could possibly be listed under this heading are indicative of the scope and diversity of computer use in health promotion today. Programs exist which will keep family medical records, log exercise workouts, chart health behavior change strategies, dictate daily weight loss diets, and even inform you of the best time to have intercourse if a couple is trying to conceive or when to avoid intercourse as a means of contraception. One diet management program (The Boston Computer Diet) helps adults set nutrition and weight control goals and then tracks the progress towards these goals. The software allows the user to choose between three different nutrition counselors who will comment on the success or failure to adhere to the individualized diet plan. One of the counselors is flexible and forgiving when the user deviates from their chosen path, another does not stand for lax behavior and will let the user know that this is not acceptable, the third is somewhere between the two.

Similar programs have been developed for people who desire to track their physical fitness activities. A major athletic footwear company has gone so far as to market a running shoe with a microprocessor in the heel to record running workouts. When the individual returns from a run s/he simply connects the shoe to a IBM or Apple computer and the software records and charts the workout.

Trends in the increasing availability of health management software combined with the sustained interest personal health enhancement may have implications for the way the public views general health care. Hawkins, Duncan, and McDermott (1986) suggest that as computers become more prevalent within our society, people will take more responsibility for their own self-care and health promotion. Rather than pay to see a professional health practitioner, users will consult with their personal computer first to see if they can do something on their own. As an example, users will be able to input symptoms of illness and vital signs (e.g., temperature, heart rate, aches and pains) and the computer will recommend whether or not it is necessary to see a physician. One group of researchers has even attempted to develop an interactive program which counsels users on solving their psychological dilemmas (Wagman, 1980; Wagman & Kerber, 1980). In many ways a computerized counselor will never be able to replace a human counselor, however, some of the limitations of this application may eventually be overcome with the development of natural language processors and expert decision systems (Sharf, 1985).

Concerns About Computer Assisted Health Promotion

The future of computer assisted health promotion holds many promises, but professionals should be cautious of this new tool, and they should warn others of its limitations. People will often view information delivered by the computer with unquestioned validity and since recommendations for personal health enhancement are often controversial, educators now and in the future will have to be increasingly wary of the appropriateness of health information transmitted by computers. For instance, when examining health promotion software one should be critical of supporting documentation that does not list references. The danger lies in people receiving inaccurate and misleading information on which to base their health decisions. The educator's job is to screen these instructional programs as they would any other educational material. Petosa and Gillespie (1984) have presented guidelines which outline some important characteristics of high quality instructional software specific to health education.

The misuse of health assessment software is an illustration of how computer use can do more harm than good. Many health professionals are concerned that persons using these programs may not be aware of how the assessment instruments are constructed and thus will not understand the limitations of applying normative data to individuals. In the case of health risk appraisals, risk factors are generally derived from studies that involve middle class, middle aged, white subjects (Fielding, 1982; Wagner, Berry, Schoenbach, & Graham, 1982). The primary limitation of HRA lies in the inadequate data base of morbidity and mortality factors for various age groups, minority and ethnic groups. The black teenage male living in the inner-city ghetto will not have the same health risk factors as the older white male living in the affluent suburbs of that city (Randolfi, Irvine, & Davis, 1986). Extrapolating data to make lifestyle recommendations for individuals is not as simple as some microcomputer programs make it seem.

As with health risk appraisals educators should be knowledgeable of how nutritional analysis programs operate. Research indicates that people will not be able to properly interpret the results of some diet analysis programs without assistance (Smiciklas-Wright, Pelican, Byrd-Bredbenner, & Shannon, 1984). As an illustration, individuals may not understand how to use RDA values (e.g., what is the significance of having 86 percent of one's RDA for selenium?). Users of these programs need to be aware that nutrition analysis programs convey a false sense of accuracy when they measure the nutrient content of foods to the nearest microgram. Students should understand that the nutritional content of foods can vary greatly depending on how the food product was grown or processed, its shelf life, refrigeration, cooking method, and cooking time (Rogan & Yu, 1984). A tendency exists to view output from nutritional analysis as precise measurements fostering unwarranted credibility. It should be understood that the results are, at best, good ballpark estimates of one's nutritional intake.

Problems may also arise if the assessment is performed incorrectly or if measurements are not precise. In this case, regardless of the how sophisticated the output looks, the results are useless. As an example, with fitness appraisal programs the accuracy and validity of these tools are more reliant¢ upon the individual conducting the fitness test than they are upon the software program. If the individual taking measurements is untrained or negligent, the client or student may be given an inappropriate exercise prescription.

As illustrated, validity and reliability are always a concern with health assessment software. This may explain why there are so few quality stress management assessment programs available. Stress is very subjective and no two people will view the same stressor in quite the same way. Therefore, the less concrete the analysis, the more open it is to interpretation.

The computer offers health promotion opportunities not available through other educational media. Software should take advantage of the computer's unique qualities as a teaching machine for health and physical education. Health and physical educators have a responsibility to be knowledgeable of any analyses being performed and should assess the quality of health information being presented by the program. The result of these efforts should ensure a better health promotion efforts and a healthier population.

References

Bailey, W. J., & Pigg, R. M. (1983). Public health education in the silicon age: The microcomputer revolution in professional preparation. *Health Education*, *14*(6), 61-64.

Bauer, L. & Surman, O. S. (1985). Microcomputer-assisted relaxation. *Perceptual and Motor Skills*, *61*, 499-502.

Bosworth, K., Gustafson, D. H., Hawkins, R. P., Chewing, B., & Day, D. (1983). Adolescents, health education , and computers: The body awareness resource network (BARN). *Health Education*, *14*(6), 58-60.

Breckon, D. J. (1985). *Microcomputer applications to health and health science*. Unpublished manuscript, Central Michigan University, Mt. Pleasant, MI

Breckon, D. J. & Pennington, R. M. (1986). Interactive videodiscs: A new generation of computer assisted instruction. *Health Values*, *10*(6), 52-55.

Cicciarella, C. C. (1983). The computer in physical education. *Journal of Physical Education, Recreation and Dance*, *54*(9), 18,32.

Deardorff, W. W. (1986). Computerized health education: A comparison with traditional formats. *Health Education Quarterly*, *13*(1), 61-72.

Fielding, J. E. (1982). Appraising the health of health risk appraisals. *American Journal of Public Health*, *72*(4) 337-339.

Hawkins, W. E., Duncan D. F., & McDermott, R. J. (1986). Can high technology make self-care a social movement? *Family and Community Health*, *9*(2), 37-45.

Hon, D. (1982). Interactive training in cardiopulmonary resuscitation. *Byte*, 108-138.

Levenson, P. M. & Morrow, J. R. (1986). A comparison of Noninteractive and interactive video instruction about smokeless tobacco. *Journal of Educational Technology Systems*, *14*(3), 193-202.

McDermott, R. J. & Belcastro, P. A. (1983). The microcomputer bandwagon is here - but watch your step! *Health Education*, *14*(6), 76-78.

National Health Information Clearinghouse (1984). *Healthfinder: health promotion software, (ODPHP, DHHS)*. Washington, D.C., U.S. Government Printing Office.

Petosa, R. & Gillespie, T. (1984). Microcomputers in health education: Characteristics of quality instructional software. *Journal of School Health*, *54*(10) 394-396.

Randolfi, E. A. (1987). Software Review: Biofeedback Microlab. *The Computing Teacher*, *14*(5) 38-41.

Randolfi, E. A. (1985). Software review: Make It Click. *The Computing Teacher*, *12*(9), 40-45.

Randolfi, E. A. (1986). *The diffusion of a curriculum innovation in higher education: A course in microcomputer applications for* health educators. Unpublished doctoral dissertation, University of Oregon, Eugene.

Randolfi, E. A., Irvine, B., & Davis, L. (1986). A course in microcomputer applications for the health educator. Family and Community Health, 9(2), 51-59.

Rogan, A. & Yu, S. (1984). Some problems with nutritional analysis software. Journal of Nutrition Education, 16(2), 65-66.

Sharf, R. S. (1985). Artificial intelligence: Implications for the future of counseling. Journal of Counseling and Development, 64(1), 34-37.

Smiciklas-Wright, H., Pelican, S., Byrd-Bredbenner, C., & Shannon, B. (1984). Clients' comprehension of a computer analyzed dietary intake printout". Journal of Nutrition Education, 16(2), 67-69.

Wagman, M. (1980). PLATO DCS: An interactive computer system for personal counseling. Journal of Counseling Psychology, 27(1), 16-30.

Wagman, M. & Kerber, K. W. (1980). PLATO DCS, an interactive computer system for personal counseling: Further development and evaluation. Journal of Counseling Psychology, 27(1), 31-39.

Wagner, E. H., Berry, W. L., Schoenbach, V. J., & Graham, R. (1982). An assessment of health hazard/health risk appraisal. American Journal of Public Health, (72), 347-352.

Woodward, J. P., Carnine, D., & Davis, L. G. (1986). Healthways: A computer simulation for problem solving in personal health management. Family and Community Health, 9(2), 60-63.

COMPUTER ASSISTED PROGRAMS FOR ASSESSING CARDIOVASCULAR FITNESS

David Bacharach
Ernest Randolfi

Department of Health and Physical Education
Syracuse University, 201 Women's Building
820 Comstock Avenue, Syracuse, NY 13244-5040

Abstract

The mounting interest of personal health and fitness has generated a large demand for overall health risk appraisals. The computer assisted programs now available give an advantage to all types of organizations and facilities as far as time and money. This survey evaluates a few computer controlled evaluation programs that are capable of determining risk of coronary artery disease from items such as age, gender, heredity, exercise habits, smoking habits, blood pressure, heart rate, body composition, and stress level inventories. Prescreening inventories have also been shown to be an effective way to determine the risk of cardiovascular disease. These screening procedures can be used to identify individuals that should not undergo a submaximal or maximal stress test without proper medical supervision, or prior to having a physician's approval. Along with these relatively simple data based tests, computers have also been used to help train medical personnel to identify exercise results. Current research has concluded that computers have been found to be remarkably accurate at predicting the potential risk of coronary artery disease using the above described criterion and that these programs are inexpensive and cost effective for schools, health agencies, and private health facilities.

Introduction

In today's society, we spend over $150 billion on our health. Staggering but true, and the mere mention of the sum is to suggest an intricate network of coordinated and organized enterprise. But how well coordinated is it? Essentially helter-skelter, the business of medicine expands with little planning. As Diamond (1983) stated, "the enormous expense has attracted the attention of economists pursing their lips and shaking their heads in confusion as the web of health care grows", (p. 219). In a health care system that is changing so rapidly, the assessment of relative costs and effectiveness quite often subject to revision. This makes them an addition to the problem rather than the solution.

The non-medical community, namely universities, health agencies, and fitness clubs have found a need for educating and testing individuals concerned about their overall health status. This mounting interest has propagated the development of computer assisted health risk appraisals. Tests which at one time were done with paper and pencil can easily and more efficiently be done by computer. Examples of simple programs of this nature are described by Newell and Skurray (1985). Although these uses are an improvement over previous methods, the capability of the computer to assist an individual in the actual fitness evaluation and/or diagnosis of serious health risks goes far beyond overall health status.

Computer Usage

A computer assisted instructional (CAI) program for interpreting electrocardiogram (EKG) changes developed by Larsen and Bacharach has been used to help educate exercise test technologists in identifying potentially dangerous EKG wave forms. More comprehensive health risk appraisals have been developed such as the one established by the Center for Disease Control in Atlanta, Georgia or the Wellness Profile authored by Bacharach and McGlade for the Texas A&M Wellness Institute. These programs typically include not only coronary risk factors, but other measures such as strength, flexibility, body composition, physical work capacity, stress level inventory, and nutrition profiles.

Saunders and Saunders (1985) found programs on nutrition, body composition, and human energy expenditure that were easy to understand and operate. Baer and Surman (1985) tested 20 subjects using a stress reduction feedback program controlled by the computer and found significant decreases in stress and anxiety as expressed by the Spielberger State Anxiety Scale. These are only a few examples of specific applications that computers can enhance. A major advantage of course, would be to have all these tests available in one package with a large data base which most microcomputers are presently capable of handling. Then individuals could easily compare themselves to national norms on a variety of topics.

In terms of medical technology, Thomas

(1974) identified three major categories of health care: nontechnology; half-technology; high technology. Initially, the large body of what Thomas refers to as "nontechnology" consists of the bedside management of patients. Basic functional needs such as eating, drinking, and normal ambulatory activity (e.g. getting in and out of bed, walking to the bathroom) are addressed in this stage. Much of this time is spent on proper education of the patient's new health status; however, from a statistical standpoint, education's ability to alter the eventual outcome of the disease is largely unknown. On the other side of the spectrum is the type of health care called "high technology" (Thomas, 1974). Stemming from the result of genuine understanding of disease mechanisms, one can see that it requires relatively little expense and is easy to deliver when compare to any other form of medical technology.

There are encompassed by these two opposite ends of the health care continuum is the level Thomas (1974) best termed "half-technology". These are measures introduced after the fact in an effort to control or lessen the disease process. It is at this tertiary treatment level that we must concern ourselves when dealing with the assessment of cardiovascular fitness and the management of coronary artery disease (CAD).

No assessment is perfect. The degree of it's competence is measured in two terms: sensitivity and specificity (Pollock, Staniloff, & Diamond; 1983). The sensitivity of a test measures the proportion of individuals with the presence of disease that are diagnosed positive by that test. Specificity is the capacity of the test to allow an individual to accurately indicate the absence of disease. As explained by Pollock et al., (1983), the Bayesian theorem used for analysis of CAD probability has become a major topic of discussion in medical literature. The Bayesian theorem approach is based on the assumption that, in well defined groups a qualitative result of any test can be expressed as a percent probability or likelihood of disease. The problem then becomes how to define a specific group of individuals.

Generally speaking, items such as age, gender, heredity, body weight, activity level, hypertension, diet, and stress are used to categorize individuals. Negative scores in a single category or a combination of several items would suggest a potential for higher risk of CAD. With heart disease being the leading cause of death in the United States today, accurately determining an individual's risk of CAD would seem important. However, Hage (1983) indicated that people coming to a physician with no apparent symptoms of CAD were more concerned with their personal appearance or body image rather than their overall health. Only those individuals who showed some degree of risk and/or concern for CAD were counseled or referred to a specialist.

Health Risk Appraisal

Let us first look at the processes involved in determining the probability of CAD. Initially, the physician must attend to the entire gambit of health. Through conversation, medical history, physical examination, and laboratory testing, they are capable of reducing the number of potential causes of the present state of condition to only a few possibilities. The contribution computers may have on this process of judgement may be uncertain. If the cognitive span during diagnosis must be reduced by rules or formulas, which is true in many instances, then the assistance of a computer program would benefit the physician (Blois, 1980). In programs such as Bleich's acid-base analysis, requests are programmed into the flow chart to obtain further information, suggest differential diagnosis, and/or suggest possible therapy for the patient. Certainly the computer can only supply the user with information that it has in it's data base; however, the amount of storage capacity and retrieval speed it may have is far beyond what the human has been capable of doing.

Since one cannot be more than 100% correct, we can see in areas other than medicine that statistical procedures such as those used by actuarians can approach and sometimes equal the performance of set standards by human predictors (Blois, 1980). As far as judgement-like processes are concerned, the ordinary problems seem to be ones in which computer assistance would not be warranted. However, as the problem becomes more defined, our ability to make accurate judgements about diagnostic tests become more and more difficult without the assistance of some statistical procedure (Blois, 1980; Diamond, 1983). These probabilities are used by the computer in the data base as well as the basis for the Bayesian theorem.

In 1979, Diamond and Forrester extended the idea of using the Bayesian theorem in cardiology by combining the pretest likelihood of CAD based on age, gender, and symptoms expressed by four tests: EKG, cardiac fluoroscopy, thalium scan, and cardiomyography. The program they devised has been available in software packages since 1980 to be used on microcomputers in the cardiologist's office. A second example of computer-assisted diagnosis of CAD was presented by Diamond, Staniloff, and Forrester in 1983. It was called the "computer-assisted diagnosis and evaluation of coronary artery disease" (CADENZA). It was designed to analyze the results of noninvasive tests that patients could perform. Over 1000 subjects were evaluated and classified according to their coronary risk factors which were a lipid profile, smoking habits, exercise, heredity, gender, age, and blood pressure. From these individuals, an index of disease severity was determined to classify others taking the same tests.

In studies by Wong, Tibits, and O'Donnell (1982) and Hlatky, Botvinik, and Brundage (1982) a comparison was made between the computer-assisted analysis and judgements of experienced cardiologists. In both cases the agreement between the two methods was highly significant. Pollock et al., (1983) indicated that a program like the CADENZA could be used as a strategy for cost effective diagnostic testing of CAD. They further suggested that this form of testing could be a valid instrument to prevent an inaccurate diagnosis of CAD for individuals being treated by a specialist rather than a cardiologist.

Current Applications

Since 1983, pretest screening using computer data bases and graded exercise tests using 12-lead EKG have become commonplace in many facilities doing complete fitness evaluations. The problem still remains however, that a large number of organizations do not have the equipment needed to do a test of this nature or qualified personnel to administer such a test. In these situations, it becomes very important to rely on a computer based fitness evaluation program. If several standard conditions must be met by each subject to be allowed to complete a submaximal or symptom-limited maximal exercise stress test, one can effectively limit the danger of testing someone that could be at high risk for CAD. Specific criterion for such tests are outlined by the American College of Sports Medicine (ACSM). These conditions are easily controlled by the computer to determine if the subject is allowed to undergo an exercise stress test without medical supervision. In this way, there is no need to have trained medical staff for subjects meeting the criterion of the program. Those that do not meet the minimum standards can be referred to their physician or to a facility that can do the testing with proper equipment and personnel.

A second application of the computer assisted fitness programs are their ability to be used for health tracking. It is very simple to maintain a running log of an individuals's daily exercise, diet, and physiological parameters such as heart rate and blood pressure. Several programs are capable of designing an exercise routine taking into consideration current daily activity, resting heart rate, and blood pressure. Besides daily tracking, a program can easily track monthly and yearly trends of all items crucially important in identifying the risk of CAD.

Naturally there is no guarantee that such computer aided activity will prevent one from having hypertension, a stroke, heart attack, or arteriosclerosis, but it has shown to be the most efficient way of identifying current risk status as well as the trend for potential problems in the future.

References

Baer, L., & Surman, O. S. (1985). Microcomputer assisted relaxation. *Perceptual and Motor Skills*, *61*, 499-502.

Blois, M. S. (1980). Clinical judgement and computers. *New England Journal of Medicine*, *303*, 192-197.

Diamond, G. A. (1983). Computer diagnosis: Revolution or revelation? *International Journal of Cardiology*, *2*, 219-220.

Diamond, G. A., & Forrester, J. S. (1979). Analysis of probability as an aid in the clinical diagnosis of coronary artery disease. *New England Journal of Medicine*, *300*, 1350-1358.

Diamond, G. A., Staniloff, H. M., & Forrester, J. S. (1983). Computer assisted diagnosis in noninvasive evaluations of patients with suspected coronary artery disease. *Journal of American College of Cardiology*, *1*, 296-308.

Hage, P. (1983). Primary care physician: First stop for exercise advice. *Physician and Sports Medicine*, *3*, 149-152.

Hlatky, M., Botvinik, E., & Brundage, B. (1983). Diagnostic accuracy of cardiologists compared with probability using Bayes' rule. *American Journal of Cardiology*, *49*, 1927-1938.

Newell, G. J., & Skurray, G. R. (1985). Computer software in physical education: What is available? *The ACHPER National Journal*, *108*, 12-14.

Pollock, B. H., Staniloff, H., & Diamond, G. A. (1983). Computer assisted interpretation of noninvasive tests for diagnosis of coronary artery disease. *Cardiovascular Review and Report*, *4*, 367-375.

Saunders, R. H., & Saunders, J. E. (1985). Computer assisted instruction as an aid for teacher of health and physical education in Queensland schools. *The ACHPER National Journal*, *108*, 4-8.

Thomas, L. (1974). *The lives of a cell*. New York: Viking Press Inc., 31-111.

Wong, D. G., Tibits, P., & O'Donnell, J. (1982). Computer assisted Bayesian analysis in the diagnosis of coronary artery disease. *Journal of Nuclear*

Medicine, 23, 83-85.

Author Notes

A representative list of health related programs is available by request to: National Information Clearing House, P.O. Box 1133, Washington, D.C. 20013

Guidelines for graded exercise testing are available by writing to: American College of Sports Medicine, P.O. Box 1440, Indianapolis, IN 46206.

AUTOMATED PLAGIARISM DETECTION: A SIMPLE APPROACH

James L. Wolfe

Indiana University of Pennsylvania
Computer Science Department
Indiana, PA 15705

Abstract: The design, implementation, and testing of a program to analyze student-written compositions for the occurrence of plagiarism are described. The program was written to be used with a system for on-line submission and grading of student compositions. The type of plagiarism detected consists of duplicated and near-duplicated sentences. Statistics and explanations are presented regarding the analysis of a large sample of student compositions. The scope, capabilities, and effectiveness of the program are described; possible additions are suggested.

INTRODUCTION

Reading, critiquing, and grading student compositions can be tedious chores for an instructor. Evaluating a student's spelling, grammar, writing mechanics, and style requires considerable concentration and effort. If an instructor attempts to look for plagiarism in the compositions at the same time, the grading and critiquing become even more complicated. Yet, plagiarism is such a serious form of academic misconduct that an instructor cannot be totally indifferent to it.

If a computer program could be devised to detect plagiarism, the instructor should welcome it (so long as the program is both effective and easy to use). By letting the computer deal with plagiarism, the instructor can concentrate on evaluating the compositions without distraction. The burden of remembering that "I've read that somewhere before" and then finding out where can be taken off the back of the instructor and given to the computer.

Previous work at Indiana University of Pennsylvania (Cross and Wolfe 1985) created an environment in which students can submit their assignments to the instructor through the computer, without the use of paper. A system called HANDIN/ONLINE allows students to submit compositions, instructors to grade them and to make comments, and students to see these comments and their grades all through the use of video terminals. As a by-product of this system, the student submissions are held in files and are readily available for additional analysis, such as plagiarism evaluation.

The Computer Science department at IUP teaches a senior seminar and a computer literacy course in which students are required to write short compositions. These compositions are essentially reviews or summaries of computer-related articles or books. The seminar students are senior Computer Science majors who generally write about technical topics. The computer literacy students are non-majors who write about computer applications in society. For two semesters, instructors in both courses required students to use the HANDIN/ONLINE system to submit their compositions. Thus, a large collection of student compositions was acquired and the opportunity to test the automation of plagiarism detection was at hand.

PLAGIARISM

Plagiarism occurs in many forms with varying amounts of effort needed to recognize each. For example, the detection of plagiarism related to writing style or plagiarism involving computer programs is quite difficult, requiring analyses of syntax and semantics. Detection of such forms of plagiarism is outside the scope of this paper. The goal here is more basic; it is to detect the simplest, most flagrant forms of plagiarism. These forms include the copying of one student's sentences or paragraphs by another student, such copying with slight alterations, and (with limited application) copying of portions of published articles by students.

To accomplish this detection, comparisons need to be made among all sentences of all student compositions. It is generally not convenient to make comparisons between the source article or book and what the student has written. The library is not on-line to this extent and such comparison would not be practical. However, experience has shown that many students (especially those in the computer literacy course) select the same articles on which to base their compositions. Consequently, plagiarism of published articles can be detected indirectly if two students "borrow" the same sentence(s) from an article.

Plagiarism is not something that an instructor expects to find in every third or fourth composition. Plagiarism occurs less frequently than that. This fact, in itself, is an argument for computerized plagiarism detection. It is difficult for an instructor to remain alert for something that occurs so seldom. A program can handle this thankless task with unrelenting vigilence.

ANALYZE

The program to detect plagiarism is called ANALYZE. It is designed to interface to the HANDIN/ONLINE system so that submissions can be analyzed without modifying them. Figure 1 shows how ANALYZE fits into the system. Students use HANDIN to "Submit" their compositions and to read the "Feedback" from the instructor. Instructors use ONLINE to "Read" and to "Critique" compositions. Instructors can use ANALYZE to "Reduce" the compositions to a more manageable form and to "Compare" compositions.

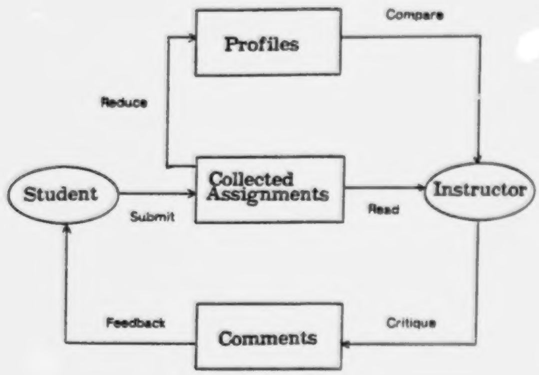


Figure 1: HANDIN/ONLINE/ANALYZE System

The overall approach to plagiarism detection and reporting within ANALYZE is to

1. reduce each student composition to a short series of numbers, called a profile,

2. compare the composition profiles (matches involving two or more profiles indicate the possible occurrence of plagiarism),

3. report the possible plagiarism in terms of locations within the submitted composition text, and

4. provide a way to view the compositions at the locations in question.

By examining the report and viewing the compositions at the indicated places, the instructor can evaluate the compositions and judge whether or not the matching sentences constitute plagiarism.

ANALYZE is designed to be used interactively and to provide reasonably fast response to the instructor. To provide this response, as well as to insure that the program is effective, the following requirements were imposed on ANALYZE.

1. Generation of profiles must be fast. To this end, hash tables and simple arithmetic operations were used in the generation.

2. Profiles must be much smaller than the compositions. A typical composition of 2 1/2 pages occupies about 5000 bytes in the submission files; a typical profile occupies about 300 bytes.

3. Comparison of profiles must be fast. Comparison is based on use of a fast sorting technique.

4. All significant plagiarism should be detected. The unit of plagiarism comparison was chosen as a sentence. Each sentence from a composition is represented in the profile. It can be argued that copying less than a sentence does not consitute plagiarism at all.

5. Harmless repetition should be ignored. Repeated references to an author, title, publisher, course name, or student name within a submission or across submissions should be disregarded.

Reductions

ANALYZE does not attempt to compare the actual text of compositions. To do so would lead to a very slow and complicated technique. Instead, compositions are compressed and represented numerically. Each student submission for each assignment is reduced to a profile. A profile consists of a pair of numbers for each sentence in the submission: one number represents the "distinguishing" words of the sentence; the other number encodes the sentence's source (a combination of assignment number, student number, and sentence position within the composition). Distinguishing words are ones that are considered important to the sentence, those that characterize it. For example in the following sentence, distinguishing words are underlined.

SOME INFORMATION ABOUT THE MEANING
OF THE SENTENCE IS USUALLY LOST.

Naturally, in selecting only distinguishing words from a sentence and then reducing these words to a single number, much information about the content, structure, and style are lost. However, sufficient information is retained so that sentences containing the same distinguishing words (in any order) are reduced to the same number.

Before describing the details of the reduction technique, the terms word and sentence as used by ANALYZE should be defined. A word is a sequence of characters ending with a blank. All letters in a word are converted to uppercase, if they are not uppercase already. Punctuation that may appear within a word (e.g., hyphen and apostrophe) and punctuation that may appear adjacent to a word (e.g., comma, colon, and semicolon) are ignored. All numbers are ignored. A sentence is a sequence of words ending with either a period or a double quote. These terminating characters were chosen so that normal English sentences and direct quotes are regarded as sentences by ANALYZE.

A sentence is reduced to a numeric value through the following algorithm.

1. For each word, perform these steps.
    a. If the word contains fewer than three letters, ignore it completely. Such small words are not regarded as distinguishing.
    b. Interpret each word of three or more letters as a base-26 integer (as if A through Z were the digits) and represent it as a 36-bit word value. Any overflow that occurs during this process is ignored.
    c. Try to find the word value in a table of insignificant words. This "stop list" table is based on the words in the negative dictionary of the SIRE system (McGill and Noreault 1977), with a few additions. The stop list is implemented as a hash table for fast searching; the table has 250 entries. Words on the stop list are not considered to contribute to the essential content of the sentence. Thus, if a word value is found in the table, the word is ignored.
2. Sum the values for all words that pass these tests to get a sentence value, again ignoring overflow. Because the sentence value is simply the sum of word values, the order of the words in the sentence is immaterial. Thus, rearranged sentences and sentences that differ only in their insignificant words still map to the same sentence value.
3. Combine the student number, course number, assignment number, and line number where the sentence starts into a single 36-bit integer that represents the position and writer of the sentence. This position value is produced with a few simple arithmetic operations.

Using 36 bits to represent a sentence value provides a large range for values and makes the probability of two distinctly different sentences generating the same sentence value very small. With more than 68 billion possible sentence values, there is less than a 1 in 100 chance of random sentences producing equal sentence values when considering a group of 20,000 sentences. ANALYZE is currently limited to working with no more than 20,000 sentences.

Comparisons

Using ANALYZE, the instructor can compare the profiles from any combination of assignments for a course. Thus, a student cannot plagiarize another student's previous assignment and go unnoticed. Such scope is essential if students make many submissions, if classes are large, and if students are allowed to choose their own topics for the submissions. All these conditions held for the Computer Science classes whose submissions were analyzed. The additional ability to compare submissions across courses is planned for the future.

ANALYZE begins its comparison by collecting the profiles (sentence and position values) for all specified submissions and sorting on the sentence values. The comparison is simple: ANALYZE looks for equality of sentence values, i.e., exact matches only. After the sort, equal sentence values will be adjacent to each other and are easy to pick out. Nothing is done with sentence values that are unequal, no matter how close the values are. The techniques used to produce the sentence values can cause very different sentences to have nearly equal sentence values.

All groups of equal sentence values (known as "matching sentences") are collected with their associated position values and are used to build a report that lists the cross-submission matches. The report specifies the students' names, assignments, and locations within the submissions where the matching sentences appear for all matches involving different submissions. Matching sentences within the same submission are counted but not considered further. Such intra-submission matches always result from harmless repetition, e.g., a repeated article title or author's name. Because a student cannot plagiarize his/her own work within a single submission, nothing more needs to be done.

Based on the report, the instructor can use ANALYZE to examine pairs of matching sentences and the text that surrounds them. The instructor specifies what is to be examined by using the information in the report. It is then possible to scroll through the two submissions in question, individually or in parallel. In this way, the instructor can evaluate the contexts in which the matching sentences occurred and decide whether or not plagiarism is involved.

FINDINGS

Figure 2 shows some basic statistics about the submission data that was used to test the ANALYZE program. The data represent assignments from two offerings of the seminar course and three offerings of the computer literacy course.

	Seminar	Literacy
Number of Students	60	178
Number of Submissions	213	456
Average length (in words)	526	362
Average number of sentences per submission	29.7	21.1
Percentage of words found to be distinguishing	49%	51%

Figure 2. Basic Statistics

Seminar students typically have five composition assignments; literacy students have three such assignments. However, not all submissions for these assignments were collected through the HANDIN/ONLINE system; this caused discrepancies between the numbers of students and the number of submissions. Literacy submissions are generally shorter and less technical than the seminar submissions, as might be expected in comparing a non-major's course with a senior, major's course. The percentages of distinguishing words are at approximately 50% for all samples. This fact is consistent with the SIRE system findings for ordinary compositions (McGill and Noreault 1977).

Figure 3 shows the numbers of sentence matches in various categories in each of the five course offerings. The number of matches represents the total number of sentences that match one or more sentences in a submission for the same course. Thus, if two sentences match, each contributes a count of one to the number of matches.

Course	Number of Sentences	Number of Matches	Percentage of Matches
Seminar 1	5036	164	3.3%
Seminar 2	3425	119	3.5%
Literacy 1	607	16	2.6%
Literacy 2	7010	65	0.9%
Literacy 3	1980	43	2.1%

Figure 3. Sentence Matches

The relatively high percentage of sentences that match in the two seminars and the first literacy offering is due to the instructor not controlling the format of the submissions. As a result, the student's name, the bibliographic reference and other standard composition features were included in the analysis. For the other two literacy offerings, the instructor required a specific format for the compositions. This reduced the number of insignificant intra-submission and cross-submission matches (Figure 4), because ANALYZE was directed to ignore portions of each submission where the bibliographic reference and student identification were required to be.

All intra-submission matches were the result of students using page headers containing their name and course number, repeated references to an article's title or its author, or other innocent repetition. The relatively high number of intra-submission matches for Literacy 3 is the result of one student who inadvertently duplicated three paragraphs within the same submission.

Course	Total Intra-Submission	Distinct Cross-Submission
Seminar 1	60	32
Seminar 2	67	19
Literacy 1	12	1
Literacy 2	13	23
Literacy 3	32	4

Figure 4. Types of Matches

The distinct cross-submission matches are the ones that represent possible plagiarism. The cross-submission numbers in Figure 4 represent unique instances, i.e., each matching-sentence group is counted once, rather than counting once for each matching sentence. Surprisingly, many of these instances have three or more sentences that match. The reason for these multiple matches can be seen in Figure 5. It lists the reasons for all of the 79 distinct cross-submission matches.

Cause of Match	Numbers of Distinct Matches Seminars	Literacy Course
Person name	34	3
Journal name	8	2
Article title	3	1
Location	0	6
Explanation	2	1
Word	3	8
Phrase	1	2
Sentence	0	5

Figure 5. Reasons for Sentence Matches

The "person names" that matched were either the article's author or the student, himself/herself. The authors' names were in the bibliographic reference; the students' names were in the page headers that the seminar students used. The "journal name", "article title", and "location" matches also have their source in the bibliographic references. Many students chose the same journal or article to write about; consequently, there were many matches for popular journals and articles. The lower numbers of person name and journal name matches for literacy submissions are the result of the instructor controlling the composition format during the Literacy 2 and 3 offerings. The instructor for these courses directed ANALYZE to ignore the lines of the submissions where the bibliographic reference was required to be.

Literacy students were required to state in their submissions the "location" where they acquired the article about which they wrote. (A frequent "location" was the library.) Seminar students had no "location" requirement and, therefore, had no such matches.

The "explanation" matches were instances where students used an acronym and then wrote it out in full. The "word" matches were the result of students using quoted terms, usually jargon, such as "bulletin board" or "phreaking" or "hackers." These words are taken directly from the articles being written about.

The instances of genuine plagiarism are represented by the eight entries in the "phrase" and "sentence" catagories. A phrase is a group of words the occurs between an ending quote and the period that ends a sentence or between the beginning of a sentence and a beginning quote. ANALYZE considers these groups of words to be sentences. Of the eight sentence and phrase matches, four are exact duplicates with respect to the words actually used in the compositions and four are slightly different because of the non-distinguishing words used. Investigation of these instances revealed that they did not result from inter-student plagiarism. All eight instances resulted from students plagiarizing a published article. In each case, two or more students had independently plagiarized the same sentence or phrase from an article. The remainder of the compositions involved in these instances were substantially different.

ANALYZE found one instance of plagiarism that is not counted in the previous statistics. One seminar student actually submitted the same composition for his first and fifth assignments, probably thinking that the instructor would not notice. Naturally, ANALYZE matched every sentence of the two submissions. One copy of this student composition was removed from the collection for which the statistics are shown.

CONCLUSIONS AND DIRECTIONS

ANALYZE does indeed detect simple plagiarism in student-written compositions. Admittedly, the program has not been tested completely; however, it has proven effective on actual compositions. It also performed far better than the instructors in detecting plagiarism. During the teaching of the courses, the instructors found only two of the instances that ANALYZE reported and none that it did not report. The instructor did not even find the instance of the student resubmitting the same composition for a second assignment.

Did ANALYZE detect all instances of simple plagiarism? No! There was at least one instance of plagiarism that was not reported; it turned up during the investigation of a sentence match. The reason that ANALYZE missed the instance is that the second occurrence of the sentence had a misspelled distinguishing word.

It was known that misspelled words would be a problem for ANALYZE before starting the project. Changing a letter in a distinguishing word results in a different word value; this leads to a different sentence value and a possible missed match. Misspellings could have been a large problem for ANALYZE if it had not been for the instructors; they required the students to use a spelling checker on their submissions. Thus, the incidence of misspellings was kept very low and ANALYZE could function effectively.

ANALYZE did not ignore all harmless repetition; however, it did quite well. The incidence of matches involving names, titles, and locations is very low. The requirement of a specific format for the literacy compositions led to far fewer matches of these types. Out of 456 literacy submissions having 9,597 sentences, only 12 name, title, or location sentence matches were found. The ideal would have been to find zero.

There are a number of adjustments that may be made to the ANALYZE program or to the way the instructor specifies assignments that can improve the reduction and comparison functions. Improvements in these functions should increase the program's effectiveness and/or efficiency in detecting plagiarism. These adjustments include

1. Making some selected additions to the stop list table. This action could result in smaller percentages of distinguishing words; however, a large change in percentages cannot be expected without using a much larger table.
2. Recognizing abbreviation periods and decimal points during reduction and distinguishing them from the period that ends a sentence. This action will make ANALYZE's definition of a sentence more similar to the common definition of an English sentence.
3. Extending the position value associated with a sentence so that it includes course information. This action will permit comparisons across courses, as well as across assignments.
4. Having the instructor place increased emphasis on the use of a spelling checker. This action will improve the quality of the compositions and let ANALYZE do a better job.
5. Having the instructor require that compositions use a specific format. The effectiveness of using this requirement in eliminating person name, journal name, and article title matches has already been shown.

The techniques used by ANALYZE are not adequate for trying to detect plagiarism in student-written computer programs. Looking for exact matches between statements in two programs would reveal only the crudest kind of program plagiarism. Experience has shown that students who plagiarize programs generally use more elaborate methods than the duplication of statements. Much more sophisticated analysis involving programming style and language syntax is needed to deal with program plagiarism. Measures of control flow and data flow (Van Verth and Ralston 1983) could form an integral part of a system to detect program plagiarism.

REFERENCES

1. Cross, J.A. and Wolfe, J.L., "Paperless Submission and Grading of Student Assignments," ACM SIGCSE Bulletin 7, 1 (March 1985), 31-33.
2. Mc Gill, M.J. and Noreault, T, "Syracuse Information Retrieval Experiment (SIRE): Rationale and Basic System Design," Syracuse University, May 1977.
3. Van Verth, P.B. and Ralston, A., "A System for Automatic Grading of Programming Style," Proceedings National Educational Computing Conference, 1983, 208-213.

IDENTIFICATION AND SELECTION OF SOFTWARE

Vicki Smith Bigham, Chair
ICCE Past President
Houston, TX

ABSTRACT

Across the United States, a variety of approaches are being used to accomplish the evaluation and selection of software appropriate for schools. Panel members will discuss some of these approaches, citing particulars from projects or endeavors on which they are currently working.

In New York City, there are 32 independent school districts. Evaluation of software is done at a central level, while selection and purchasing is done at the local level. The overall plan requires enough flexibility to allow for meeting each district's philosophy and needs, whether it be computer-assisted instruction, a Logo focus, or one of a number of others. Evaluation is accomplished for 250-400 titles each month, with the cumulative list now numbering more than 10,000. The Software Coordinator from the New York City Board of Educator's Office of Technology will describe the system, which allows something for everyone.

Over the past three years, California has devoted a large amount of resources for evaluating technology-based programs, in order to positively affect the choices educators are making in selecting software and video programs and to eventually improve the quality of programs developed. While these efforts have proven to be effective, a broader set of strategies is being planned. The Director of the Office of Educational Technology of the California State Department of Education will focus on efforts to date, lessons learned, and strategies for the future, including software development and instructional materials adoption policy.

In 1986, the Texas State Board of Education established a software advisory committee. Composed of classroom teachers and other school district employees, the committee has been charged with responsibilities to: (1) develop and implement a system under which the group continuously evaluates software; (2) make recommendations to the Board concerning software that should be approved and acquired; and (3) cooperate with software publishers and designers in subsequent development. A former member of the Software Advisory Group and current Educational Specialist with the Division of Educational Technology at the Texas Education Agency will discuss the Software Advisory Group's commitment to technology-based education, the powers and duties of the group, subcommittees and goals formed, and the current status of efforts.

In July of 1984, the Southeastern Regional Council for Educational Improvement Technology Advisory Committee recommended and received approval to explore the feasibility of developing a multistate software evaluation process. The Software Evaluation Exchange Dissemination (SEED) Project is a collaborative effort coordinated by the Southeastern Educational Improvement Laboratory and the state education departments in Alabama, Florida, Georgia, Louisiana, Mississippi, North Carolina, and South Carolina. The SEED software evaluation process has been designed and implemented by staff members of the participating state agencies. The Louisiana representative to the SEED Task Force will describe the willingness of educators in these southeastern states to share the fruits of their efforts with one another.

Panel members will discuss the benefits and problems arising from approved software lists and the control or influence of such efforts on software development.

PANELISTS

Tom Boudrot
Texas Education Agency
Austin, TX

Jeff Branzburg
New York City Board of Education
New York, NY

Wendy Harris
California State Department of Education
Sacramento, CA

Shirley McCanless
Louisiana Department of Education
Baton Rouge, LA

'SHIFTING SANDS' IN THE UNDERGRADUATE COMPUTER SCIENCE CURRICULUM

Allen B. Tucker, *Moderator*
Colgate University

ABSTRACT

Significant recent movement has occurred in the evolution of computer science curricula in colleges and universities, and away from the traditional model described in the ACM Curriculum 78. In 1984 and 1985, for instance, two ACM committees developed new standards for the introductory courses CS1 and CS2. In 1986 a Model Curriculum in computer science was developed that suggests a whole new place for the roles of theory and mathematics in an undergraduate computer science degree program. Significant new ideas have evolved around the roles of software engineering and laboratories in undergraduate computer science education. What is the substance of these changes, and what are their desired outcomes? What is the probable impact on the Advanced Placement Curriculum in Computer Science for secondary schools? The panelists will address these issues in the following way:

Rogers:	the Introductory Courses (CS1 and CS2) and the Advanced Placement Curriculum
Gibbs:	the role of Software Engineering in the curriculum
Kelemen:	the role of Laboratories in the curriculum
Tucker:	the role of Theory in the curriculum

A lively question and answer exchange will be encouraged throughout this panel session.

PANELISTS

Norman E. Gibbs
Carnegie-Mellon University

Charles F. Kelemen
Swarthmore College

Jean B. Rogers
Stanford University

Allen B. Tucker
Colgate University

KEYBOARD CONNECTIONS: USING FrEdWRITER IN THE CLASSROOM

Greg Butler
Miller Computer Education Centre
Miller, New South Wales, Australia

ABSTRACT

Many schools throughout the world have started to use the *FrEdWriter* word processing package in their classrooms. It offers teachers many advantages over other commercial word processing packages. The most obvious is price, *FrEdWriter* being available through the public domain. Many resources have been, or are in the process of being developed to support the use of *FrEdWriter* in the classrom. *FrEdWriter* is fast becoming one of the most widely used word processing packages in schools.

But how good is this word processing package for use in schools? And are teachers using it to its potential?

This presentation will look at how teachers may use *FrEdWriter* with their students. It will consider how *FrEdWriter*, along with *Keyboard Connections*, may be used in the development of reading/writing skills. It will show how *FrEdWriter* may be incorporated in a variety of classroom applications and teaching styles.

Some of the many suitable classroom applications of this exciting piece of software, which teachers may develop, will be discussed and demonstrated.

For those ready to take on the challenge, more advanced *FrEdWriter* applications will also be presented. These include using *FrEdWriter* as a teaching aid, a desktop publishing tool, a simple data base, an aid in teaching spelling, a writing and grammer checking tool, a thought processor, and for creating interactive adventure stories. Examples of these classroom applications will be demonstrated.

All applications are simple and can be easily included in junior and senior classrooms. *FrEdWriter* has already managed to get a foot into many classrooms. After this prsentation many more teachers may open the door wide and say "Come inside."

FrEdWRITER, FrEdSENDER, AND TELEPHONES: A PRACTICAL APPROACH TO NETWORKING ACTIVITIES

Al Rogers
Region 15 TEC Center
San Diego County Office of Education
San Diego, CA

ABSTRACT

Telecommunications in the classroom has thus far been at best a questionable activity in terms of positive benefits in the development of basic skills. It has been a theoretical, impractical, difficult, expensive, or irrelevant activity in most situations. Probably the best that can be said of telecommunications in a popular sense is that it is a TOOL IN SEARCH OF A TASK.

Researchers at the University of California in San Diego have investigated the role of telecommunications in the academic life in the classroom. In their studies, networking was seen as an extension of the writing process, as an aspect of the need to consider and define "audience." They found that students who do meaningful writing to a meaningful audience write better than students who write for any other purpose. They also showed that the real-world audience, when combined with the nature of the telecommunications medium itself, allows language experiences not easily available through other means.

All successful networking experiences studied contained the following essential ingredients:

1. There is a task to be accomplished;
2. Using telecommunications to accomplish the task must be easier or more effective than other means;
3. Participants in the project know each other;
4. Participants are committed to the task; and
5. There is some degree of accountability in performing the task.

These findings have guided the development of the San Diego School-Net Project. This informal, grass-roots telecommunications networking project centers around *FrEdWriter*, *FrEdSender*, and CMS (Computer Mail System)....readily available, low-cost programs for Apple computers. The project is developing specific lessons and activities to foster the development of real-world audiences and generate the kinds of teacher and student collaboration that will improve student language and communications skills. The San Diego School-Net project currently has more than 25 bulletin board systems in California, Nebraska, New Hampshire, Connecticut, New York, Pennsylvania, and Puerto Rico. EMail and student writing is exchanged daily between these sites.

This session, by the author of *FrEdWriter* and co-developer of *FrEdSender* and CMS, describes the San Diego School-Net network and provides illustrations and samples of some of the activities and experiences of a practical, functional school-networking project.

APPLIED MATH, FIELD SCIENCE, AND PIAGET

M. Heidi Imhof
University of Oregon
Eugene, OR

ABSTRACT

This session will explore an interdisciplinary approach to "doing science," appropriate for middle school/junior high students. We have seen many articles advocating active, hands-on science programs, and are left to wonder how to do this in our own classrooms. This model is designed with a progression of activities that begin in the classroom and lead to the "field." The initial classroom activities reinforce the prerequisite math and science skills that, applied together, form the framework for student understanding. Field techniques will include use of a simple clinometer, compass, transect lines, point-centered quadrants, etc. Participation will be the mode as we work through data collection techniques, then bring them "back into the classroom" for analysis--using data bases and spreadsheets. The activities in this session are geared for middle school/junior high teachers. Come ready to participate!

COMPUTERS IN A HIGH SCHOOL
SCIENCE/MATH/COMPUTER SCIENCE MAGNET

Michael Haney
Blair Management Program
Silver Spring, MD

Lola Piper
Blair Management Program
Silver Spring, MD

ABSTRACT

Computers have many possible roles in high schools, but few of them are realized. Although they are seen as possible tutors, tools for problem solving, or portals to vast information systems, to fulfill these promises means a major revision of the curriculum. To capitalize fully on their power, a world rich in diversity and unconstrained by historical approaches to education must be created.

The Blair Magnet Program is a high school program that began with educators rethinking what we teach and why we teach it--more appropriately, what children should learn and how they should learn it. A completely new computer science curriculum was written beginning with a one-year survey course in grade 9 and ending with a variety of semester excursions in grades 11 and 12. More importantly, computers became the glue that fastly tied mathematics and science to computer science, and a mutual dependence grew. From that relationship, many interdisciplinary experiences arose--experiences where students freely used computers to help solve math problems, compose music, design parts, simulate processes, and run and monitor experiments as well as model the results.

This talk will relate just several of the many student experiences which integrate separate disciplines. (Others are related in the handouts.) Each is presented as an example with an emphasis on the process of planning and carrying out such experiences. They are part of a four-year high school curriculum which, through deliberate design and extensive planning, integrates courses at every level without compromising any discipline.

The first example comes early in the ninth grade. The students do a small interfacing experiment which combines the use of an air track, photogates, a computer and spreadsheets. They then merge their results into a larger spreadsheet for statistics class. There they generate a curve of best fit for physics where they are studying uniform acceleration. Each teacher reinforces the lessons from other classes; and, through a common experience, each teacher coordinates the students' efforts, covering different objectives.

Another example comes in grade 10, when students are studying satellite photos of star fields in earth science. The satellite transmits images of star fields in green, red and blue. In their Pascal class, the students write a program to merge three digital files, each of which represents the same star field taken in a different color. (Spin-offs include digitizing photos.)

A third example relates to biology, where students use a wide variety of application software to conduct research, organize their ideas and notes, and develop tables, charts and illustrations. This session will discuss how the students' biology project builds on skills introduced in the freshman computer science course, emphasizing DIALOG, Chem Abstracts, and other information retrieval systems.

A final example is the culminating activity of the Logo unit simulating projectile motion. After an introductory lesson in simulations, the roles simulations play in a variety of fields, and simulation languages, students are presented with a (physics) projectile motion problem. They design and implement the solution in Logo.

These were just a few of many examples. Implementing such overlap between courses requires the will, many hours of effort, and the time to know what your colleagues are doing. It also takes flexibility and the willingness to alter sequences to make concurrent experiences more timely and more meaningful for children. The Blair Magnet has been lucky because its program began from scratch with no assumptions about the validity of current curriculum practices. Consequently, it has become a laboratory rich with educational experiences and lessons for us all. Some ideas are worth adapting and adopting, others are not. For the 19 educators who will participate in this project, it has become an opportunity to try those ideas that we always felt would benefit children, if only we had had the chance to try them.

ON-LINE SEARCHING FOR A BIOLOGY RESEARCH PROJECT

Erica K. Lodish
Blair Management Program
Silver Spring, MD

Goria A. Seelman
Blair Management Program
Silver Spring, MD

ABSTRACT

The Biology Research Project is an interdisciplinary assignment coordinated by a biology teacher, a research and experimentation teacher, and a media specialist. The project objectives require the student to integrate knowledge of on-line data base searching techniques, traditional research skills, biological terminology, problem solving, experimental design, and communication skills.

The students are introduced to search strategies in MAGAZINE INDEX, BIOSIS PREVIEWS, AGRICOLA, AQUACULTURE, MEDLINE, ENVIROLINE, EMBASE, SCISEARCH and other data bases after they have had a lesson in topic selection and an overview of the project requirements. The media specialist adapted a search request form to help students define their problems and identify concepts and terminology necessary to execute a search.

Students are required to do at least two layers of searches: one for background information and a second for information to establish an experimental design. The background information provides students with clarification of terminology, related subjects, comparative research, and clues to unanswered questions on their subjects. Once the students clarified their problem and discovered the variables to be tested, this became their experimental design. Their second search sought articles regarding establishing controls, isolating a single factor to study, data collection techniques, and methods for evaluation.

To fulfill another project objective, students must include an original bibliographic source. The original source may be a letter, telephone call, interview, or seminar with an authority on the students' research topic. Students sometimes are able to locate mentors when searching on SCISEARCH by looking for citations whose corporate source is in the Washington, D.C., area. Another resource that they use to find contacts is The Information Connection Bank, created by a joint committee of the Montgomery County Volunteer Services representatives and VITRO Laboratories. The Information Connection is a local data base of speakers, field trips, mentors, shadowing, summer opportunities, scholarships, etc. Both of these data bases cite local authorities in the subject area queried. The authorities in The Information Connection Bank have volunteered to assist students as mentors.

On-line searching exposes the student to varied problem solving techniques, selected terminology, and the determination of the hierarchy of concepts. It also provides students with a direct route to contacts for information and advice.

APPLICATIONS

David Miles, Thomas Flynn
James TenEyck
Nick Gordon, Bob Latzer, Jill Lewis
Ronald D. Schwartz

ABSTRACT: Developing Student Computer Literacy with Lap-top Computers

David Miles, Thomas Flynn, Ohio University Belmont, St. Clairsville, Ohio 43950

Purpose

This report will discuss the planning, implementation, and results of program designed to improve computer literacy skills at a two-year, public, regional campus. Students were loaned lap-top computers for the duration of their involvement in the program. This program began in 1985 and continues into its second year.

Goals

1. To overcome the artificial constraints of the computer lab and to enable students to have convenient 24-hour computer access.
2. To select a two-year series of regular curriculum courses that would facilitate students' mastery of basic computer applications: word processing, spreadsheet, telecommunications and data bases.
3. To study the program to determine students' response and use.

Results

1. The laptop has proved to be easily used in the classroom, library and home.
2. The two-year program was not feasible. Even at the freshman level, it was impossible to find a common sequence of courses that would meet students' major requirements and give an opportunity for the use of computer skills. Consequently, the plan was revised to a one-year sequence.
3. Questionnaires show students spending between three and four hours each week outside of class working on their computers. Students also assert that as the course progresses, they feel more strongly that the computer assists them in mastering their course material.

ABSTRACT: Extending The Classroom Environment with Communications Tools

James TenEyck, Department of Computer Science, Marist College, Poughkeepsie, New York 12601

The use of communications tools such as computer mail and class newspapers can facilitate the interaction between diverse yet related classes. These tools permit significant communication between classes with a minimum of coordination by the faculty involved. I will describe a project using these methods.

My class in Computers and Society this spring considered the reality of a computer-communications revolution and discussed the possible implications for society of such an event. This course benefited from the involvement of faculty and students from other disciplines. The use of communications tools allowed a loosely structured interaction between classes without any major modification in the syllabus or method of delivery by the involved faculty. Students joined interest groups that included students and faculty from other courses. These groups communicated through the mainframe memo facility, and each interest group was responsible for discussions and written presentations on their topic. A student- edited newspaper was distributed weekly in each of the participating classes. The students were expected to submit articles, appropriate book reviews, letters and other material to the class newspaper. Students thus received advice from faculty and other students within the interest group during the development of their topic, and shared their final effort with classmates in the extended classroom.

My report will discuss four aspects of this project: planning, coordination, implementation, and evaluation of the results. Only the impact of the project upon the Computers and Society class will be considered.

ABSTRACT: Computers Across The Curriculum: The First Year

Nick Gordon, Bob Latzer, Jill Lewis, Jersey City State College, Jersey City, New Jersey 07305

The project trains faculty and students to use the computer as an academic tool in order to increase the amount and quality of writing and reasoning in General Studies courses. Three-week workshops are being given over the summer to familiarize faculty with various computer applications programs, to teach them how to use writing as a learning tool, and to help them develop strategies for using the computer to improve instruction in writing and critical thinking in all disciplines. Follow-up includes faculty seminars in the semester following the summer training. Results of the summer workshops and follow-up seminars are being disseminated to all faculty at the college. A Computers Across the Curriculum Lab staffed by faculty throughout the day and evening has been set up to support anticipated increased writing and computer use. Students will be required to take a one-credit course called Computer as Tool, which teaches them the same computer applications that the faculty is learning in the summer workshops. Evaluation of the project includes studies of whether there is a change in the amount of writing and computer use in General Studies courses.

The computer program we are using is *AppleWorks*, which includes a wordprocessor, data base, and spreadsheet. In our presentation we will discuss ways the faculty are using *AppleWorks* in their workshops and how we are managing the other components of this complex and interesting project.

ABSTRACT: Computer Instruction Within the Schools of Business Administration

Dr. Ronald D. Schwartz, Wayne State University, School of Business Administration, Detroit, Michigan 48202

The need for business students to learn the fundamentals of "application" or "productivity" software has emerged so rapidly that the majority of curricular offerings in the schools of business administration have not kept pace. Associated with these needs are a multitude of related concerns, which must be properly addressed if the initial problem is to be completely resolved.

Several years ago the sole intent of the introductory computer course for business students was to provide an understanding of (1) the components and functions of the computer, (2) the logical processes involved in computer problem solving, and (3) the appropriate coding instructions for solving business applications.

Within the last few years, however, the usage of application software has dramatically altered the manner in which professionals perform their daily tasks. This is evidenced by the number of business men and women who depend upon spreadsheets to analyze data or managers who employ data base management programs to extrapolate segments of information from a data base. It is now readily apparent that the users' needs have been altered. Greater emphasis has now shifted from learning the fundamentals of a higher level programming language to exposure to and usage of application software.

This researcher will recommend issues and curriculum modifications that must be addressed and then resolved if the schools of business administration are to keep pace with the evolving needs of the "end user" in computer education.

K-12 INTEGRATION

Dave Moursund
Jean Dames, Ann Dana, Michael Warner
Gerard Berry, Francine Gallagher, James Phifer, Ronald Zirkle

ABSTRACT: Effective Inservice for Computer-Integrated Instruction

Dr. Dave Moursund, CATE, 1787 Agate St., University of Oregon, Eugene, OR 97403

The instructional uses of computers can be divided into three main categories: computer science, computer assisted instruction, and computer-integrated instruction (CII). This research project concerns CII.

The focus of this three-year National Science Foundation project is to:

1. Develop a research-based model and support materials for effective inservice CII;
2. Test and replicate the model with upper elementary and secondary math, science, and social science teachers; and
3. Train a number of inservice providers to use the inservice model and support materials.

The project is developing four "notebooks," one for each major group of participating teachers. These notebooks will be distributed in both hardcopy and machine readable form. They will contain detailed information on how to conduct an inservice, the materials needed, extra materials for use by teachers in their schools, and an analysis of the research supporting this inservice model.

The project is nearing completion of its second year. An eight-session (16-hour), discovery-based model for inservice education has been developed and used with upper elementary teachers and secondary math and science teachers.

The discovery-based model for inservice education is based on careful examination of the literature and on many years of the project staff's inservice experiences. Extensive formative evaluation has helped shape the inservice format and materials. The results should interest inservice providers who help educators learn about use of the computer-as-a-tool in elementary and secondary schools.

ABSTRACT: The Changing Roles of the School District Computer Coordinator

Jean Dames, Technology Consultant, North Cook Educational Service Center, Glenview, Illinois; Ann Dana, Microcomputer Consultant/Teacher, Elementary District 181, Hinsdale, Illinois; and Michael P. Warner, Cadre Services Coordinator, DuPage-Kane Educational Service Center, Glen Ellyn, Illinois

A small group of individuals--self-appointed, board dictated, or district created computer coordinators--have accepted the responsiblility for promoting and maintaining the microcomputers in the schools. The computer coordinator's role in the schools has evolved as quickly as the uses of computers themselves.

How has this role changed since its inception in the early 1980s? How are the services of the state-established Educational Service Centers used? To answer these and other questions about the evolution of the position of computer coordinator, 165 surveys were sent to K-12 school districts in four northern Illinois regions.

For about one third of those responding a computer coordinator position was created; while for nearly one half, the responsibilities were added on to their existing positions--coordinators of media, learning centers, testing, gifted, data processing; directors of curriculum and instruction, public relations, career ed; and teachers of computer science, physics, social studies and math.

Due to the diversity of the responses, a follow-up questionnaire was sent and an in-depth interview was conducted with a selected sample of coordinators in the four northern Illinois regions. This gave more specific information on the role of the computer coordinator and answered such questions as: How do you establish support groups? How do you handle the change in attitudes of teachers and students? And, how do you keep up with the latest breakthroughs in technology? Again, these answers showed an interesting variety of responses.

The changing role of the computer consultants and their place in the mainstream of our educational systems affects teachers, students, and the communities they service.

ABSTRACT: Using Technology to Improve the Learning of Problem Solving, Science, and Writing Skills

Gerard J. Berry, Francine L. Gallagher, James D. Phifer, Ronald E. Zirkle, Office of Instructional Technology, Fairfax County Public Schools, Fairfax, Virginia

Fairfax County Public Schools in Virginia has spent several years developing a model program demonstrating ways to improve the teaching of skills in problem solving, science, and writing through the use of computers in the classroom.

Funded by the U.S. Department of Education, the program, Project LITT (Learning Improvement Through Technology), was designed with specialized training courses, summer institutes, and curriculum workshops to train teachers and prepare materials. The program focused on the use computers as tools to improve the teaching of problem solving skills in grades K-4 and 7-8, science skills in grades 9-12, and writing skills in grades K-12. The problem-solving components used the computer languages Logo and BASIC along with specially developed software. The science component used commercial software and microcomputer based laboratory tools. The writing component emphasized a four-phase writing process and the use of word processors. The program was implemented and evaluated at 31 sites.

This program is a positive example of an effective method for introducing and promoting the use of technology with instruction. The presentation will include a description of the program, unique training opportunities made available to teachers, key elements in its development, and its effects on instruction in problem solving, science and writing.

INTEGRATING EDUCATIONAL SOFTWARE INTO HOME AND SCHOOL LEARNING

P. Kenneth Komoski
EPIE Institute
Watermill, NY

Wells Hively
EPIE Institute
Teachers College
Columbia University
New York, NY

ABSTRACT

During the last decade, there has been an increasing use of educational software in schools and homes. However, in most communities this use has been of a tentative and/or sporadic nature. During the last few years, there has been an increasing amount of rhetoric calling for a more coherent and comprehensive approach to the use of educational software in the school curriculum. In a few cases, this rhetoric has been turned into action (e.g., California's Technology in the Curriculum (TIC) Project and Apple's recent curriculum correlations of highly rated software.)

While such activities are encouraging signs, they are efforts related to one state and one computer company. And although these efforts seem to be stimulating other similar activities, such activities seem to be failing to face two major realities:

1. Computer software needs to be related both to a school's curriculum goals and objectives and to all the other materials (especially textbooks) that are being used on a day-to-day basis by teachers and learners.
2. There are about 10 times the number of computers in the homes of school children than there are in their schools, and the largely untapped educational potential of this formidable computer resource needs to be related effectively (integrated with) to in-school educational computing.

During the last few years, EPIE Institute has developed two major program efforts aimed at helping educators and parents to deal more effectively with these two realities.

The first of these efforts is the development of the Integrated Instructional Information Resource (IIIR), a relational data base designed to help schools use a locally designed curriculum as the basis for selecting and interrelating textbooks, computer software, and other institutional resources into a well-integrated, well-aligned curriculum program.

The second of these efforts is the Classroom/Community Connections Program, a project aimed at helping schools to work with parents to help their children make effective use of at-home computing in ways that integrate at-home learning experiences to the school curriculum.

EPIE's agenda for the future is helping schools, parents and community agencies to use the IIIR Data base to bring about a productive educational relationship between the increasing amounts of educational software being produced each year and improved in-school and at-home learning.

A COGNITIVELY MOTIVATED APPROACH TO TEACHING COMPUTER PROGRAMMING

Elliot Soloway, Chair
Department of Computer Science
Yale University
New Haven, CT

ABSTRACT

Significant progress has been made in cognitive science toward understanding how people solve problems, learn, etc. For example, novices and experts have been extensively studied in fields such as physics and medicine. My research group, the Cognition and Programming Project, has focused on the task domain of computer programming. Our goal has been to understand how novices and experts design, write, read, debug, etc., computer programs. For example, we have devoted considerable energy to studying novice programmers. We have analyzed the bugs they make and have attempted to construct theories that can explain the origins of the bugs. Based on our understanding of the bugs made by students and how experts go about comprehending programs, we have built and classroom tested PROUST, a system that can identify the nonsyntactic bugs in students' programs for a class of moderately complex, introductory programming assignments. PROUST's performance is on par with that of a human teaching assistant. Finally, we have attempted to make explicit the knowledge underlying programming -- the programming specific knowledge as well as the problem solving knowledge that is used in programming.

Given this considerable body of research into programming and programmers, we feel ready to move into a prescriptive mode and make suggestions as to how introductory programming and problem solving can be more effectively taught. Thus, the goals in the talk are: (1) to convey a sense of the data and theory that has motivated our work; and (2) to make concrete suggestions as to how programming and problem solving can be effectively taught. That is, I don't believe it is enough to simply give a list of prescriptions for teaching; such lists appear as rabbits from a hat. Rather, I think that enough of the theoretical underpinnings and data can be readily presented so that attendees can see where the prescriptions come from and how they fit together as a coherent whole.

PANELISTS

Jean Rogers
Stanford University
Stanford, CA

Jim Greer
University of Texas, Austin
Austin, TX

NOT TO BE TAKEN OUT OF CONTEXT: RESEARCH ON COMPUTERS IN THE CLASSROOM

Dr. Betty Collis
University of Victoria
Victoria, B.C., Canada

ABSTRACT

Educators are continually looking for useful and helpful guidelines regarding the impact of computers in the classroom. Practitioners and policy makers often inquire about the impact and potential benefits of computers in the learning process, and researchers look for generalizable effects relating to computer use in educational settings. Literally thousands of research studies investigating the impact of computers in education have been conducted since 1980 alone. If we attempt to synthesize the results of even a small portion of this research or, more generally, to better understand the effects and benefits of computers in education, we must reconcile ourselves to dealing with a very complicated situation.

The classroom context, in particular, exerts a powerful influence on what happens when students use computers. Even slight variations in the classroom context in which computer use takes place can make a major impact on the overall effectiveness of a computer-related learning experience. We contend that educational computer use cannot be studied in any meaningful way without considering the surrounding context in which the use occurs. Following Chomsky's perspective, we see classroom computer use as context-embedded, rather than context-free, and we see this occurring even in the CAI situation where the student is meant to interact independently with a school computer for drill and practice or with tutorial software.

The following variables, all relating to the classroom context, can be identified as potential influences on the use and impact of computers in the learning process:

Teacher-related: Awareness of computer applications; expertise with computer applications; general organizational skills; communication skills and overall teaching effectiveness; andpreferred teaching style.

Instruction-related: Degree to which computer use relates to overall instructional unit or experience; curriculum requirements of the particular course; expectations for student achievement; and degree of support and organization provided for computer use.

Situational: Location of computers and access to school computers; available hardware and software resources; school policy for computer usage; and school expectations for student and teacher behavior.

Class Environment: Size of class; age; ability level; SES; previous experience of class with computers; homogeneity/heterogeneity of class; idiosyncratic variables in class situation; and familiarity of class with variations in classroom routine.

Student-related: Ability level; preferred learning style; attitudes toward computers; peers' attitudes toward computers; susceptibility to influence peers; self-confidence with computers; and motivation to achieve.

Other: Quality of software; ergonomic considerations; and novelty of computer use.

In this presentation, we will discuss specific examples of the influence these variables have on the learning process in the context of three key areas in computer applications: Logo, word processing, and simulation programs. Research studies will be cited that demonstrate how a particular classroom variable can be strongly related to the quality of the computer usage that occurs. We will discuss the implications of this classroom-variable/computer-use interaction for the teacher, the school or district policy maker, the researcher, and the developer of educational software and associated instructional materials.

Traditional research on CAI attempts to minimize the influence of contextual variables, but as efforts are increasingly made to integrate computers into classroom programs, we will need to develop ways to study this integration when the computers are deeply embedded in the instructional environment. The classroom teacher, who knows the classroom environment better than anyone, will be an important partner in this research.

PANELISTS

Dr. Decker Walker
Stanford University
Stanford, CA

Wayne Grant
Stanford University
Stanford, CA

NATIONAL PROGRAMS TO IMPROVE EDUCATION THROUGH TECHNOLOGY

Andrew R. Molnar, Chair
National Science Foundation
Washington, DC

ABSTRACT

Dr. Susan Chipman will present a brief history of the Office of Naval Research's involvement in research on Intelligent-Computer-Assisted-Instruction (ICAI) and describe a large number of on-going ICAI projects. She will also discuss other ONR interests in cognitive research and its relation to ICAI.

Dr. Conrad Katzenmeyer will describe the activities of the Office of Educational Research and Improvement and the Educational Technology Center at Harvard University. The Center focuses on mathematics and science and ways of learning with computers. He will also discuss OERI's planning exercises on technology education and discuss questions related to the levels and types of programs a new focus might take

Dr. Andrew R. Molnar will describe the activities of the National Science Foundation program in the Applications of Advanced Technologies. The Program is concerned with issues at the forefront of technological applications in science and mathematics education. The Program currently supports research and development of knowledge-based systems and intelligent tutors in calculus, algebra, geometry, pre-algebra and algorithmic problem solving. It is supporting the development of computational tools and symbol manipulation systems for problem solving and advanced applications. It also supports work on authoring systems and on programming and problem solving.

PANELISTS

Susan Chipman
Office of Naval Research
Arlington, VA

Conrad Katzenmeyer
Office of Educational Research and Improvement
U.S. Department of Education
Washington, DC

SPONSOR: ICCE

AUTHOR INDEX

Abernathy, Lucky J. 279
Abramson, Gertrude W. 273, 318
Abramson, Theodore 318
Adams, Eileen 8
Aiken, Robert J. 283
Aiken, Robert M. 3, 137
Albright, Harry 162
Alo', Richard A. 236
Anderson-Inman, Lynne 155
Arney, David C. 283
Bacharach, David 360, 366
Bacon, J. Roger 108
Bacon, Linda 213
Balajthy, Ernest 213
Balestri, Diane 277
Barclay, Tim 99, 158
Bartik, Betty Jean 229
Bartley, John A. 14
Batke, Peter 230
Bear, George G. 286
Becker, Henry Jay 231
Beil, Shirley J. 64
Bender, Evelyn 20
Bennett, Randy E. 157
Bergey, Nancy Lee 10
Berry, Gerard J. 384
Berryman, Rich 229
Betts, Francis M., III 89, 206
Bigham, Vicki Smith 183, 237, 376
Billings, Karen 45, 319
Bitter, Gary G. 132, 160
Black, John B. 348
Blomeyer, Robert L., Jr. 77
Bodek, Matthew 135
Borack, Jules 182
Bork, Alfred 154
Boudrot, Tom 376
Bowers, C. A. 114
Brand, Julie A. 318
Branzburg, Jeff 376
Brennan, Joyce R. 23
Brett, George 159
Bright, George W. 77
Brownstein, Saul 229
Buck, Marguerite M. 16
Bureau, W. Edward 14
Burkholder, Leslie 27
Burnett, Tom 95
Burns, Mike 45
Butler, Greg 10, 378
Bybee, Rodger W. 204
Cannings, Terence R. 157
Carlson, Elizabeth Uzdavinis 24
Carnine, Douglas 235
Carnow, Gary A. 20
Carter, Richard 22, 48
Char, Cynthia 95
Charp, Sylvia 154
Chinitz, Paul 229
Chipman, Susan 389
Choate, Jon 158
Clark, Joe 95
Clark, Milford 48
Clarkson, David 158
Clements, Douglas H. 47, 227
Cline, Hugh F. 157
Cloughly, Rick 229
Coe, Mary Ann 12
Collis, Betty 319, 388
Commito, Ann E. 283
Conn, Margery 6
Cook, Laverne 232
Corica, Tim 237
Cory, Sheila 319
Couch, Jacqualine 324
Coulter, Neal S. 137
Cousins, Laura 229
Csongor, Julianna 14
Culp, George H. 60
Cunniff, Nancy 168
Cunningham, Beverly 298
Cunningham, Lee 298
D'Amore, Michael 26
Dames, Jean 384
Dana, Ann 384
Dannenberg, Roger 27
Davis, Lynne M. 278
Dean, Linda 14
DelForge, Clarence 48
Dennis, Verl 162
Digranes, Jo Lynn Autry 233
DiLullo, John 16
Dixon, Victoria 236
Dudley-Marling, Curt 228
Duin, Ann Hill 283
Dunlop, David 2
East, J. Philip 23
Edwards, Carol 291, 341
Ellis, James D. 204
Ellis, Margaret "Kit" 324

Epler, Doris M. 2
Evans, Paul 154
Feldscher, Jerry 229
Feurzeig, Wallace 22
Fisher, Francis Dummer 154
Fitzgibbons, Brian 11
Flach, Elisabeth G. 238
Fletcher, J. D. 63
Flynn, Thomas 382
Foley, Timothy J. 60
Forman, George 18
Forsythe, Charles 2
Foster, Leon 233
Fox, Helen 12
Francisco, Mont 45
Frankel, Fred 229
Franklin, Thomas E. 190
Frankson, Raymond 229
Frederick, Betz 130
Frederick, Harold 130
Freyd, Pamela 3, 342
Friedland, Mark 45
Friel, Susan 47
Fuller, Robert G. 60
Furlong, Mary 24
Gallagher, Francine L. 384
Galletta, Anne 318
Garrett, Mary 303
Gibbs, Norman E. 377
Gilbert, Steven W. 277
Gist, Thomas E. 60
Goldberg, Fred 318
Goode, K. Wayne 248
Goodson, Bobby 282
Gordon, Florence S. 149
Gordon, Nick 382
Gordon, Sheldon P. 38
Gore, Kay 204
Graeber, Janet 24
Grant, Wayne 388
Greer, Jim 387
Halfon, Paula 10
Hampel, Paul 6
Haney, Michael 380
Hannah, Larry 160, 321
Harrigan, Audrey 318
Harris, Judith 227
Harris, Thomas R. 206
Harris, Wendy 376
Harvey, Francis A. 259
Hasby, Patricia 213
Haslam, Elizabeth L. 89
Haugland, Susan W. 18
Hazen, Margaret 159
Heller, Eric S. 18
Heslop, Betsy 10
Hiriak, Rosemary 10
Hirshman, Joanruth 11
Hively, Wells 386
Hodges, Julia E. 137
Holmes, Glyn 230
Houston, Nancy P. 228
Hunter, Beverly 46, 96
Hydrick, C. Jane 204
Ide, Nancy M. 27, 159
Imhof, M. Heidi 379
Iverson, Joanne W. 229
Jacobson, Michael J. 60
Jobe, Holly 10
Johnson, Patricia G. 108
Jones, Randall L. 230
Jongejan, Tony 282, 322
Jordan, Delores 8
Joseph, Suzanne 24
Kahn, Jessica 3
Kahn, Ted 96
Kaplan, Carol A. 318
Katzenmeyer, Conrad 389
Kean, Michael H. 182
Kelemen, Charles F. 377
Kerrigan, John 2
Kiernan, Gerard 279
Kimmel, Howard 279
Klafter, Marcia B. 11
Kleiman, Glenn 47
Klinzing, Dene G. 18
Knauer, John 228
Koltun, Arnold 279
Komoski, P. Kenneth 386
Kosel, Marge 183
Koziara, Edward Clifford 119
Kramer, Arlene 3
Kuhn, Tom 232
Kulesa, Frank 217
LaClaire, Lafe 143
Land, Michael 253
Larkin, Jill 28
Lasker, Harry 232
Lathrop, Ann 183
Latson, Jon B. 103
Latzer, Bob 382
Lawrence, Sister Rose, IHM 5
Lawson, John D. 258
Lee, Cedrick 229
Lee, M. C. 139
Levin, Mel 8
Levine, Jacqueline Jordan 291
Lewis, Jill 382

Lewis, Sharon Johnson 182
Lieberman, Lynn M. 285
Lindemann, Erika 159
Linerode, Philip M. 53
Lodish, Erica K. 381
Longo, Stephen 229
Lopez, Robert J. 34
Lorenzen, Gary L. 60
Luehrmann, Arthur 46
Luoma, Richard 14
Mackall, Phil 233
Malian, Ida 24
Mandinach, Ellen 158
Manganiello, Anthony J. 283
Marchionini, Gary 268
Marcus, Stephen 46, 62
Marx, Bodie 183
Masters, Howard 229
Mathies, Bonnie 282
Mathis, Judi 323
McBride, Brother Mark 5
McBride, James R. 182
McCanless, Shirley 376
McEnerney, Paul 14
McGarvey, Jim 229
McGrath, Diane 302
McKillip, William D. 47
McManus, John F. 157
McQuade, Marilyn K. 60
Meadors, Joseph R. 206
Mecklenberger, James 95
Melmed, Arthur 154
Melstein, Marilyn 11
Mendez, Juan S. 318
Mensing, Linda 200
Meyer, Anne 200, 309
Miles, David 382
Moberg, Thomas 138
Modla, Virginia B. 12
Moll, Richard G. 69
Molnar, Andrew R. 389
Montgomery, Judith 22
Montgomery, Judith 206
Morris, Sandra 232
Moursund, Dave 160, 282, 384
Nelson, Mark 204
Norris, Cathleen 258
O'Shea, Mark 279
Olds, Henry F. 97
Olson, David M. 23, 175
Orndorff, Joseph 208
Owston, Ronald D. 228
Pagni, David 47
Panto, Paula J. 204
Parsegian, Elsa V. 236
Patrick, Diane 268
Perry, Linda J. 108
Peterson, Douglas L. 283
Phifer, James D. 384
Piper, Judy 20
Piper, Lola 380
Pollack, Richard 95
Poncz, Sarah 206
Popyack, Jeffrey L. 73
Pratt, Jane 98
Psihountas, George 279
Pugsley, Dale 324
Pullman, Howard W. 236
Randolfi, Ernest A. 283, 360, 366
Rawitsch, Don 320
Reisman, Fredricka 89
Richards, John 22
Rine, David C. 331
Roberts, Nancy 99, 158
Robinson, Stephanie 341
Roditi, Bethany N. 309
Rogers, Al 378
Rogers, Jean B. 258, 377, 387
Romano, Joanne 8
Rosecky, Marion H. 5
Roth, R. Waldo 138
Rothman, Paula 10
Rowe, Mary Budd 158
Rue, William P. 283
Russell, Susan Jo 341
Saterfiel, Thomas H. 182
Saylor, Beverly 156
Scheftic, Carol 28
Scherer, Charlotte 160
Schooler, Lael J. 143
Schroer, Bernard J. 248
Schwartz, Edward N. 279
Schwartz, Ronald D. 382
Scott, Thelma 98
Sebrechts, Marc M. 143
Seelman, Gloria A. 381
Shade, Daniel D. 18
Sheets, Richard 132
Sheffer, Nola 22
Shelly, Anne 335
Sherman, David M. 125
Sibert, Ernest 335
Simpkins, Alice 138
Sinkinson, Michael 12
Smith, John B. 159
Smith, Jordan M. 27
Smith, Peter 138
Smoliar, Jerry 229

Snelbecker, Glen 3
Soloway, Elliot 143, 387
Sperber, David 48
Springsteen, JoAnn 18
Stager, Gary S. 353
Stas, Shirley 45
Stecher, Brian 157
Stein, Debra K. 228
Stone, Antonia 233, 341
Strudler, Neal 85
Summers, Marguerite K. 23
Sumner, Steven 45
Swan, Karen 348
Swanson, Richard E. 60
Takacs, Helen C. 137
Taylor, Harriet G. 23, 258
Taylor, Robert P. 168, 348
Teague, Jerry 268
TenEyck, James 382
Thomas, J. D. 324
Thomas, Lajeane G. 326
Tonik, Al 229
Troutner, Joanne 204
Trowbridge, David 28
Tucker, Allen B. 377
Turner, A. Joe 137
Turner, Sandra V. 279
Vaille, John A. 281
Varden, Stuart 45
Viau, Elizabeth Anne 184
Vicino, Frank 182
Walker, Decker 388
Walkington, Pat 45
Walsh, Patricia 8
Warner, Michael P. 384
Warren, Zab 237
Watt, Daniel H. 48, 157, 227
Watt, Molly 48, 157, 227
Weaver, David 50
Weiner, Anne 135
Weller, Martha H. 60
Wenn, Richard 96, 341
White, Charles S. 331
Wideman, Herbert H. 228
Willoughby, Stephen S. 47
Wolfe, James L. 370
Wootson, Alice 8
Wresch, William 163
Yan, Chiou-Shuang 119
Yocam, Keith 98
Zirkle, Ronald E. 384
Zoromski, Ronald 190
Zurhellen, Kathleen 302